

特開平11-220655

(43) 公開日 平成11年(1999) 8月10日

(51) Int.Cl.⁶ 識別記号

H 0 4 N 5/268
H 0 4 H 1/00
H 0 4 J 3/00
H 0 4 N 7/24

F I

H 0 4 N 5/268
H 0 4 H 1/00 A
H 0 4 J 3/00 M
H 0 4 N 7/13 Z

審査請求 未請求 請求項の数27 O L (全 40 頁)

(21) 出願番号 特願平10-282609

(22) 出願日 平成10年(1998)10月5日

(31) 優先権主張番号 特願平9-271275

(32) 優先日 平9(1997)10月3日

(33) 優先権主張国 日本 (J P)

(71) 出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72) 発明者 田原 勝己

東京都品川区北品川6丁目7番35号 ソニー株式会社内

(72) 発明者 安田 幹太

東京都品川区北品川6丁目7番35号 ソニー株式会社内

(72) 発明者 根岸 慎治

東京都品川区北品川6丁目7番35号 ソニー株式会社内

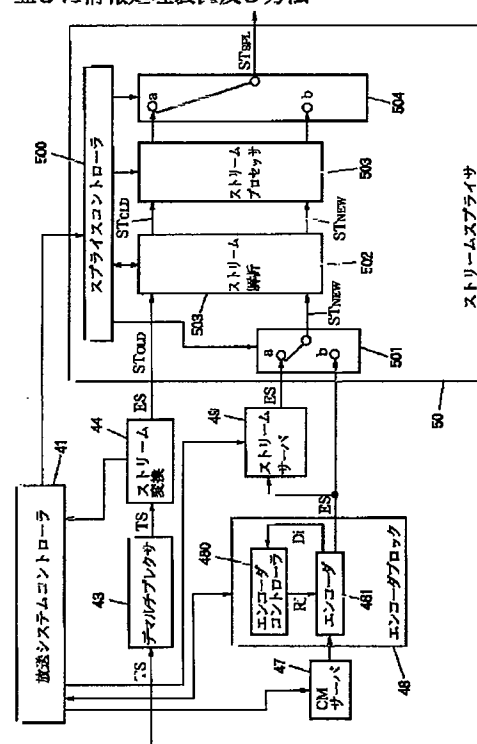
(74) 代理人 弁理士 稲本 義雄

(54) 【発明の名称】 符号化ストリームスプライシング装置及び符号化ストリームスプライシング方法、符号化ストリーム生成装置及び符号化ストリーム生成方法、並びに情報処理装置及び方法

(57) 【要約】

【課題】 シームレスなスプライスドストリームを生成するためのストリームスプライシング装置を実現する。

【解決手段】 本局からトランスポートストリームとして伝送されてきたオリジナル符号化ストリーム STOLD のストリーム形態が変換されたエレメンタリーストリームは、ストリーム解析回路 502 に供給される。ストリーム解析回路 502 は、オリジナル符号化ストリーム STOLD のシンタックスを解析して、vbv_delay、repeat_first_field 及び top_field_first 等のデータエレメントを抽出し、それらをスプライスコントローラ 500 に供給する。スプライスコントローラ 500 は、これらのオリジナル符号化ストリーム STOLD に関するデータエレメントをストリームプロセッサ 503 に供給する。



【特許請求の範囲】

【請求項1】 第1の符号化ストリームと第2の符号化ストリームとをスプライシングポイントにおいてスプライシングする符号化ストリームスプライシング装置において、

上記第1の符号化ストリームのシンタックスを解析することによって、上記第1の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、

上記ストリーム解析手段によって得られた上記第1の符号化ストリームの符号化パラメータに基づいて、上記スプライシングポイントにおいて上記第1の符号化ストリームと上記第2の符号化ストリームとがシームレスに接続されるように、上記第2の符号化ストリームの符号化パラメータを変更し、上記第1の符号化ストリームと上記符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシング手段とを備えたことを特徴とする符号化ストリームスプライシング装置。

【請求項2】 上記ストリーム解析手段によって抽出される符号化パラメータは、

vbv_delayであることを特徴とする請求項1記載の符号化ストリームスプライシング装置。

【請求項3】 上記スプライシング手段は、上記第2の符号化ストリームにおけるスプライスポイント後の最初のピクチャのvbv_delayの値を、上記第1の符号化ストリームにおけるスプライスポイント後の最初のピクチャのvbv_delayの値に書き換えることを特徴とする請求項2記載の符号化ストリームスプライシング装置。

【請求項4】 上記スプライシング手段は、上記第2の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量が、上記書き換えられたvbv_delayの値に対応するビット量となるように、上記第2の符号化ストリームにスタッフィングバイトを挿入することを特徴とする請求項3記載の符号化ストリームスプライシング装置。

【請求項5】 上記スプライシング手段は、上記スタッフィングバイトのデータ量を、上記第1の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量、及び上記第2の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量に基づいて演算することを特徴とする請求項4記載の符号化ストリームスプライシング装置。

【請求項6】 上記スプライシング手段は、上記スタッフィングバイトのデータ量を、上記第1の符号化ストリームのスプライスポイントにおけるVBVバッファのデータ占有量と、上記第2の符号化ストリームのスプライスポイントにおけるVBVバッファのデータ占有量とに基づいて演算することを特徴とする請求項5記載の符号化ストリームスプライシング装置。

【請求項7】 上記ストリーム解析手段によって抽出される符号化パラメータは、

repeat_first_fieldであることを特徴とする請求項1記載の符号化ストリームスプライシング装置。

【請求項8】 上記スプライシング手段は、上記第1の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第1の符号化ストリームにおけるスプライスポイント前のピクチャのrepeat_first_fieldの値、又は、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのrepeat_first_fieldの値を変更することを特徴とする請求項7記載の符号化ストリームスプライシング装置。

【請求項9】 上記ストリーム解析手段によって抽出される符号化パラメータは、

top_field_firstであることを特徴とする請求項1記載の符号化ストリームスプライシング装置。

【請求項10】 上記スプライシング手段は、上記第1の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第1の符号化ストリームにおけるスプライスポイント前のピクチャのtop_field_firstの値、又は、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのtop_field_firstの値を変更することを特徴とする請求項9記載の符号化ストリームスプライシング装置。

【請求項11】 第1の符号化ストリームと第2の符号化ストリームとをスプライシングポイントにおいてスプライシングする符号化ストリームスプライシング方法において、

上記第1の符号化ストリームのシンタックスを解析することによって、上記第1の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、上記ストリーム解析ステップによって得られた上記第1の符号化ストリームの符号化パラメータに基づいて、上記スプライシングポイントにおいて上記第1の符号化ストリームと上記第2の符号化ストリームとがシームレスに接続されるように、上記第2の符号化ストリームの符号化パラメータを変更し、上記第1の符号化ストリームと上記符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシングステップとを備えたことを特徴とする符号化ストリームスプライシング方法。

【請求項12】 上記ストリーム解析ステップによって抽出される符号化パラメータは、

vbv_delayであることを特徴とする請求項11記載の符号化ストリームスプライシング方法。

【請求項13】 上記スプライシングステップは、

上記第2の符号化ストリームにおけるスプライスポイント後の最初のピクチャのvbv_delayの値を、上記第1の符号化ストリームにおけるスプライスポイント後の最初のピクチャのvbv_delayの値に書き換えることを特徴とする請求項12記載の符号化ストリームスプライシング方法。

【請求項14】 上記スプライシングステップは、上記第2の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量が、上記書き換えられたvbv_delayの値に対応するビット量となるように、上記第2の符号化ストリームにスタッフィングバイトを挿入することを特徴とする請求項13記載の符号化ストリームスプライシング方法。

【請求項15】 上記スプライシングステップは、上記スタッフィングバイトのデータ量を、上記第1の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量、及び上記第2の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量に基いて演算することを特徴とする請求項14記載の符号化ストリームスプライシング方法。

【請求項16】 上記スプライシングステップは、上記スタッフィングバイトのデータ量を、上記第1の符号化ストリームのスプライスポイントにおけるVBVバッファのデータ占有量と、上記第2の符号化ストリームのスプライスポイントにおけるVBVバッファのデータ占有量とに基いて演算することを特徴とする請求項15記載の符号化ストリームスプライシング方法。

【請求項17】 上記ストリーム解析ステップによって抽出される符号化パラメータは、repeat_first_fieldであることを特徴とする請求項11記載の符号化ストリームスプライシング方法。

【請求項18】 上記スプライシングステップは、上記第1の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第1の符号化ストリームにおけるスプライスポイント前のピクチャのrepeat_first_fieldの値、又は、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのrepeat_first_fieldの値を変更することを特徴とする請求項17記載の符号化ストリームスプライシング方法。

【請求項19】 上記ストリーム解析ステップによって抽出される符号化パラメータは、top_field_firstであることを特徴とする請求項11記載の符号化ストリームスプライシング方法。

【請求項20】 上記スプライシングステップは、上記第1の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第1の符号

化ストリームにおけるスプライスポイント前のピクチャのtop_field_firstの値、又は、上記第2の符号化ストリームにおけるスプライスポイント後のピクチャのtop_field_firstの値を変更することを特徴とする請求項19記載の符号化ストリームスプライシング方法。

【請求項21】 第1の符号化ストリームと第2の符号化ストリームとをスプライシングポイントにおいてスプライシングすることによってスプライスド符号化ストリームを生成する符号化ストリーム生成装置において、上記第1の符号化ストリームのシンタックスを解析することによって、上記第1の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、上記ストリーム解析手段によって得られた上記第1の符号化ストリームの符号化パラメータに基いて、上記スプライシングポイントにおいて上記第1の符号化ストリームと上記第2の符号化ストリームとがシームレスに接続されるように、上記第2の符号化ストリームの符号化パラメータを変更し、上記第1の符号化ストリームと上記符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシング手段とを備えたことを特徴とする符号化ストリーム生成装置。

【請求項22】 第1の符号化ストリームと第2の符号化ストリームとをスプライシングポイントにおいてスプライシングすることによってスプライスド符号化ストリームを生成する符号化ストリーム生成方法において、上記第1の符号化ストリームのシンタックスを解析することによって、上記第1の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、上記ストリーム解析ステップによって得られた上記第1の符号化ストリームの符号化パラメータに基いて、上記スプライシングポイントにおいて上記第1の符号化ストリームと上記第2の符号化ストリームとがシームレスに接続されるように、上記第2の符号化ストリームの符号化パラメータを変更し、上記第1の符号化ストリームと上記符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシングステップとを備えたことを特徴とする符号化ストリーム生成方法。

【請求項23】 データを受信する受信手段と、上記受信手段で受信したデータの一部を、他のデータと入れ替える入れ替え手段と、上記他のデータを符号化する符号化手段と、上記受信手段で受信したデータを、上記他のデータと整合が取れるように変更する変更手段とを備えることを特徴とする情報処理装置。

【請求項24】 上記受信したデータと、他のデータにおけるtop_field_firstとrepeat_fieldの値を、予め決定された値にすることを特徴とする請求項23に記載の情報処理装置。

【請求項25】 上記符号化手段で符号化されたデータを保存する保存手段をさらに備えることを特徴とする請

求項24に記載の情報処理装置。

【請求項26】 上記保存手段は、予め設定された時間単位で符号化されたデータを保存することを特徴とする請求項25に記載の情報処理装置。

【請求項27】 データを受信する受信ステップと、上記受信ステップで受信したデータの一部を、他のデータと入れ替える入れ替えステップと、上記他のデータを符号化する符号化ステップと、上記受信ステップで受信したデータを、上記他のデータと整合が取れるように変更する変更ステップとを含むことを特徴とする情報処理方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】この発明は、デジタル放送システムにおいて使用される符号化ストリームスプライシング装置及び符号化ストリームスプライシング方法、符号化ストリーム生成装置及び符号化ストリーム生成方法、並びに情報処理装置及び方法であって、特に、2つの符号化ストリームをストリームレベルでスプライシングすることによってシームレスなスプライズドストリームを生成する符号化ストリームスプライシング装置及び符号化ストリームスプライシング方法、符号化ストリーム生成装置及び符号化ストリーム生成方法、並びに情報処理装置及び方法に関するものである。

【0002】

【従来の技術】図1は、現在のテレビジョン放送システムを説明するための図である。現在のテレビジョン放送システムにおいて、各家庭にテレビ番組を配信するための放送局は、全国規模のテレビ番組を制作する本局（キーステーション又はメインステーション）SKと、地方特有のテレビ番組を作成する本局系列の複数の地方局（支局）SA、SB、及びSCとから構成されている。本局SKは、全国共通のテレビ番組を作成し、その作成したテレビ番組を地方局に伝送するための放送局であって、地方局は、本局から局間伝送によって送られて来たオリジナルテレビ番組、及びそのオリジナルテレビ番組一部を地方特有向けに編集したテレビ番組の両方の番組を地方の家庭に配信するための放送局である。例えば、図1に示されるように、地方局EAは、放送エリアEA内の家庭に伝送するテレビ番組を作成する局であって、地方局EBは、放送エリアEB内の家庭に伝送するテレビ番組を作成する局であって、地方局ECは、放送エリアEC内の家庭に伝送するテレビ番組を作成する局である。尚、この各地方局において行われる編集処理とは、例えば、本局から送られてきたニュース番組に、地方独自の天気予報のプログラムを挿入したり、映画やドラマ等の番組に、地方向けのコマーシャルを挿入したりする処理のことである。

【0003】図2は、各地方局における編集処理を説明するための図であって、図2（A）は、本局において制

作されたオリジナルテレビ番組PGOLDを示し、図2

（B）は、地方局において制作された地方向けの差し替えテレビ番組PGNEWであって、図2（C）は、地方局において編集されたテレビ番組PGEDITを示している。図2に示された編集処理の例は、本局から伝送されてきたオリジナルテレビ番組のうち、コマーシャルCM1、プログラム2、及びコマーシャルCM3を、地方局において地方向けに制作されたコマーシャルCM1'、プログラム2'、及びコマーシャルCM3'に置き換える編集処理の例である。この地方局における編集処理の結果、図2（C）に示されるように、本局において生成されたテレビ番組（プログラム1、CM2、プログラム3C、及びプログラム4）と地方局において生成されたテレビ番組（コマーシャルCM1'、プログラム2'、及びコマーシャルCM3'）とが混在する地方向けのテレビ番組が生成される。

【0004】近年、現在のテレビジョン放送システムはアナログのベースバンドのテレビジョン信号を各家庭に配信するアナログ放送であるため、これらのアナログ方法システムを、デジタル技術を使用した次世代の放送システムに置換えようという試みがなされている。このデジタル放送システムは、MPEG2（Moving Picture Experts Group Phase2）等の圧縮符号化技術を用いてビデオデータやオーディオデータを圧縮符号化し、その符号化されたストリームを地上波や衛星波を利用して各家庭や他局に伝送するシステムである。特に、このデジタル放送システムとして提案されている放送技術の中で、ヨーロッパにおいて次世代の放送方式として提案されているDVB（Digital Video Broadcasting）規格が最も有力であって、このDVB規格がデファクトスタンダードとなりつつある。

【0005】次に、図3を参照してMPEG規格を用いて、ビデオデータとオーディオデータを含んだプログラムを送信側システムから受信側システムに伝送する一般的なデジタル伝送システムに関して説明する。

【0006】一般的なデジタル伝送システムにおいて、伝送側システム10は、MPEGビデオエンコーダ11と、MPEGオーディオエンコーダ12と、マルチプレクサ13とを備え、受信側システム20は、デマルチプレクサ21と、MPEGビデオエンコーダ22と、MPEGオーディオデコーダとを備えている。

【0007】MPEGビデオエンコーダ11は、ベースバンドのソースビデオデータVをMPEG規格に基いて符号化し、その符号化したストリームをビデオエレメンタリストリームESとして出力する。MPEGオーディオエンコーダ12は、ベースバンドのソースオーディオデータAをMPEG規格に基いて符号化し、その符号化したストリームをオーディオエレメンタリストリームESとして出力する。マルチプレクサ13は、MPEGビデオエンコーダ11とMPEGオーディオエンコーダ1

2から、それぞれビデオエレメンタリストリームとオーディオエレメンタリストリームを受取り、それらのストリームをトランスポートストリームパケットの形態に変換し、ビデオエレメンタリストリームを含んだトランスポートストリームパケットとオーディオエレメンタリストリームを含んだトランスポートストリームパケットを生成する。さらに、マルチプレクサ13は、ビデオエレメンタリストリームを含んだトランスポートストリームパケットとオーディオエレメンタリストリームを含んだトランスポートストリームパケットとが混在するように、それぞれのトランスポートストリームパケットを多重化し、受信システム20に伝送されるトランスポートストリームを生成する。

【0008】デマルチプレクサ21は、伝送路を介して伝送されたトランスポートストリームを受取り、ビデオエレメンタリストリームを含んだトランスポートストリームとオーディオエレメンタリストリームを含んだトランスポートストリームパケットとに分離する。さらに、デマルチプレクサ21は、ビデオエレメンタリストリームを含んだトランスポートストリームパケットからビデオエレメンタリストリームを生成するとともに、オーディオエレメンタリストリームを含んだトランスポートストリームパケットからオーディオエレメンタリストリームを生成する。MPEGビデオデコーダ22は、デマルチプレクサ21からビデオエレメンタリストリームを受取り、このビデオエレメンタリストリームをMPEG規格に基いてデコードし、ベースバンドのビデオデータVを生成する。MPEGオーディオデコーダ22は、デマルチプレクサ21からオーディオエレメンタリストリームを受取り、このオーディオエレメンタリストリームをMPEG規格に基いてデコードし、ベースバンドのオーディオデータAを生成する。

【0009】さて、従来のアナログ放送システムをこのようなデジタル伝送システムの技術を使用してデジタル放送システムに置き換えようとした場合、本局から地方局に向けて伝送されるテレビ番組のビデオデータは、MPEG2規格に基いて圧縮符号化された符号化ストリームとなる。従って、地方局において本局から伝送されたオリジナル符号化ストリームの一部を、地方局において制作された符号化ストリームに置換えるための編集処理を行なうためには、この編集処理の前に、一端、符号化ストリームをデコードして、ベースバンドのビデオデータに戻さなければいけない。なぜなら、MPEG規格に準じた符号化ストリームに含まれる各ピクチャの予測方向は、前後のピクチャの予測方向と相互に関連しているので、ストリーム上の任意の位置において符号化ストリームを接続することができないからである。もし強引に2つの符号化ストリームをつなげたとすると、符号化ストリームのつなぎめが不連続になってしまい、正確にデコードできなくなってしまうことが発生する。

【0010】従って、図2において説明したような編集処理を実現するためには、本局から供給されたオリジナルの符号化ストリームと、地方向けに制作された符号化ストリームの両方を一端デコードし、それぞれをベースバンドのビデオ信号に戻すデコード処理と、2つのベースバンドのビデオデータを編集してオンエア用の編集されたビデオデータを生成する編集処理と、編集されたビデオデータを再び符号化して、符号化ビデオストリームを生成するという符号化処理とを行なわなくてはならない。しかしながら、MPEG規格に基く符号化/復号化処理は100%可逆の符号化/復号化処理ではないので、復号化処理及び符号化処理を繰り返すたびに画質が劣化してしまうという問題があった。

【0011】そこで、近年では、供給された符号化ストリームを復号化処理せずに、符号化ストリームの状態のまま編集することを可能にする技術が要求されるようになってきた。尚、このように符号化されたビットストリームレベルで、異なる2つの符号化ビットストリームを連結し、連結されたビットストリームを生成することを「スプライシング」と呼んでいる。つまり、スプライシングとは、符号化ストリームの状態のまま複数のストリームを編集及び接続することを意味する。

【0012】しかしながら、このスプライシング処理を実現するためには以下のような2つの問題点がある。

【0013】まず、第1の問題点について説明する。

【0014】上述したMPEGビデオエンコーダ11及びMPEGビデオデコーダ22において使用されているMPEG規格では、符号化方式として双方向予測符号化方式が採用されている。この双方向予測符号化方式では、フレーム内符号化、フレーム間順方向予測符号化および双方向予測符号化の3つのタイプの符号化が行われ、各符号化タイプによる画像は、それぞれIピクチャ(intra coded picture)、Pピクチャ(predictive coded picture)およびBピクチャ(bidirectionally predictive coded picture)と呼ばれる。また、I、P、Bの各ピクチャを適切に組み合わせ、ランダムアクセスの単位となるGOP(Group of Picture)が構成される。一般的には、各ピクチャの発生符号量は、Iピクチャが最も多く、次にPピクチャが多く、Bピクチャが最も少ない。

【0015】MPEG規格のようにピクチャ毎にビット発生量が異なる符号化方法では、得られる符号化ビットストリーム(以下、単にストリームとも言う。)をビデオデコーダにおいて正確に復号化して画像を得るためには、ビデオデコーダ22における入力バッファ内のデータ占有量を、ビデオエンコーダ11で把握していなければならない。そこで、MPEG規格では、ビデオデコーダ22における入力バッファに対応するバッファとして‘VBV(Video Buffering Verifier)バッファ’という仮想バッファを想定し、ビデオエンコーダ11では、

VBVバッファを破綻、つまりアンダフローやオーバーフローさせないように、符号化処理を行なうように定義されている。例えば、このVBVバッファの容量は、伝送される信号の規格に応じて決められており、メインプロファイル・メインレベル(MP@ML)のスタンダードビデオ信号の場合であれば、1.75Mビットの容量を有している。ビデオエンコーダ11は、このVBVバッファをオーバーフローやアンダフローさせないように、各ピクチャのビット発生量をコントロールする。

【0016】次に、図4を参照して、VBVバッファについて説明する。

【0017】図4(A)は、本局において制作されたプログラム1とコマーシャルCM1とを含んだオリジナルテレビ番組を、ビデオエンコーダで符号化したオリジナルストリームSTOLDと、そのオリジナルストリームSTOLDに対応するVBVバッファのデータ占有量の軌跡を示す図である。図4(B)は、地方向けに制作されたコマーシャルCM1の部分に差し替えられるコマーシャルCM1'を、地方局のビデオエンコーダで符号化した差し替えストリームSTNEWと、その差し替えストリームSTNEWのVBVバッファのデータ占有量の軌跡を示す図である。尚、以下の説明において、本局から支局に伝送されたオリジナルテレビ番組を符号化したストリームの一部が支局において作成された新たなストリームによって置き換えられるので、このオリジナルテレビ番組を符号化したオリジナルストリームを、古いストリームであることを示す'STOLD'と表現し、オリジナルストリームSTOLDの一部に新たに差し替えられる差し替えストリームを'STNEW'と表現する。図4(C)は、スプライスポイントSPにおいてオリジナルストリームSTOLDに対して差し替えストリームSTNEWをスプライシングすることによって得られたスプライドストリームSTSPL(spliced stream)と、そのスプライドストリームSTSPLのVBVバッファのデータ占有量の軌跡を示す図である。

【0018】尚、図4において、VBVバッファのデータ占有量の軌跡において、右上がり部分(傾き部分)は伝送ビットレートを表し、垂直に落ちている部分は各ピクチャの再生のためにビデオデコーダがデコーダバッファから引き出すビット量を表している。ビデオデコーダがこのデコーダバッファからビットを引き出すタイミングは、デコーディングタイムスタンプ(DTS)と呼ばれる情報によって指定される。なお、図4においてI、P、Bは、それぞれIピクチャ、Pピクチャ、Bピクチャを表している。

【0019】オリジナル符号化ストリームSTOLDは本局のビデオエンコーダで符号化されたストリームであって、差し替えストリームSTNEWは地方局のビデオエンコーダで符号化されたストリームであるので、オリジナ

ル符号化ストリームSTOLDと差し替えストリームSTNEWとは、それぞれのビデオエンコーダで全く関係無く符号化されたストリームである。よって、地方局のビデオエンコーダは、本局のビデオエンコーダのVBVバッファのデータ占有量の軌跡を全く知らずに、独自に差し替えストリームSTNEWを生成するための符号化処理を行っている。スプライスポイントSPにおけるオリジナルストリームSTOLDのVBVバッファのデータ占有量VBVOLDと、スプライスポイントSPにおける差し替えストリームSTNEWのVBVバッファのデータ占有量VBVNEWとは異なってしまう。

【0020】つまり、スプライドストリームSTSPLのスプライスポイントSPの前後において、VBVバッファのデータ占有量の軌跡が不連続にならないようにするためには、スプライドストリームSTSPLにおける差し替えストリームSTNEWのVBVバッファのデータ占有量の初期レベルは、VBVバッファのデータ占有量VBVOLDにしなければいけない。その結果、図4に示したように、オリジナルストリームSTOLDのVBVバッファのデータ占有量VBVOLDよりも、差し替えストリームSTNEWのVBVバッファのデータ占有量VBVNEWの値が小さい場合には、スプライドストリームSTSPLにおける差し替えストリームSTNEWの部分で、VBVバッファがオーバーフローしてしまう。また、逆に、オリジナルストリームSTOLDのVBVバッファのデータ占有量VBVOLDよりも、差し替えストリームSTNEWのVBVバッファのデータ占有量VBVNEWの値が大きい場合には、スプライドストリームSTSPLにおける差し替えストリームSTNEWの部分で、VBVバッファがアンダーフローしてしまう。

【0021】次に、第2の問題について説明する。

【0022】MPEG規格に基いて符号化されたストリームのヘッダには、符号化情報を示すさまざまなデータエレメントやフラグが記述されており、これらのデータエレメントやフラグを使用して符号化ストリームを復号化するようになされている。

【0023】図2に示されたオリジナルテレビ番組の本編を構成するプログラム1、プログラム2、プログラム3及びプログラム4は、ビデオカメラ等によって撮影された29.97Hz(約30Hz)のフレームレートを有するNTSC方式のテレビジョン信号とは限らず、24Hz(毎秒24コマ)のフレームレートを有する映画素材からテレビジョン信号に変換された信号であることもある。一般的に、このように24Hzの映画素材を29.97Hzのテレビジョン信号に変換する処理を、オリジナル素材における2個のフィールドを、所定のシーケンスで3個のフィールドに変換する処理を含むことから、'2:3プルダウン処理'と呼んでいる。

【0024】図5は、この2:3プルダウン処理を説明するための図である。図5において、T1からT8は2

4 Hzのフレーム周波数を有する映画素材のトップフィールドを示し、B1からB9は24 Hzのフレーム周波数を有する映画素材のボトムフィールドを示している。さらに、図5において示された楕円及び三角形は、トップフィールドとボトムフィールドから構成されるフレームの構造を示している。

【0025】具体的には、この2:3プルダウン処理において、24 Hzのフレーム周波数を有する映画素材(8個のトップフィールドT1~T8、及び8個のボトムフィールドB1~B8)に、ボトムフィールドB2を繰り返すことによって生成されたりピートフィールドB2'、トップフィールドT4を繰り返すことによって生成されたりピートフィールドT4'、ボトムフィールドB6を繰り返すことによって生成されたりピートフィールドB6'、及びトップフィールドT8を繰り返すことによって生成されたりピートフィールドT8'の4つのリピートフィールドを挿入する処理が行われる。その結果、この2:3プルダウン処理によって、24 Hzのフレーム周波数を有する映画素材から29.97 Hzのフレーム周波数を有するテレビジョン信号が生成される。

【0026】MPEGエンコーダにおいて、2:3プルダウン処理されたテレビジョン信号は、ビデオエンコーダにおいてそのまま符号化処理されるのではなく、2:3プルダウン処理されたテレビジョン信号からリピートフィールドを除去してから、符号化処理が行われる。図5に示した例では、2:3プルダウンされたテレビジョン信号から、リピートフィールドB2'、T4'、B6'及びT8'が除去される。このように符号化処理前にリピートフィールドを除去する理由は、このリピートフィールドは、2:3プルダウン処理時に挿入された冗長なフィールドであって、圧縮符号化効率を向上させるために削除したとしても何ら画質劣化が発生しないからである。

【0027】また、MPEG規格においては、符号化ストリームをデコードする際に、フレームを構成する2つのフィールドのいずれかのフィールドを繰り返すことによって、リピートフィールドを生成するか否かを示す「repeat_first_field」というフラグを記述することを定義付けている。具体的には、MPEGデコーダは、符号化ストリームをデコードする際に、符号化ストリーム中のフラグ「repeat_first_field」が「1」の場合にはリピートフィールドを生成し、符号化ストリーム中のフラグ「repeat_first_field」が「0」の場合にはリピートフィールドを生成しないという処理を行なう。

【0028】図5に示した例の場合には、トップフィールドT1とボトムフィールドB1とから構成されるフレームを符号化したストリームの「repeat_first_field」は「0」であって、トップフィールドT2とボトムフィールドB2とから構成されるフレームを符号化したストリームの「repeat_first_field」は「1」であっ

て、トップフィールドT3とボトムフィールドB3とから構成されるフレームを符号化したストリームの「repeat_first_field」は「0」であって、トップフィールドT4とボトムフィールドB4とから構成されるフレームを符号化したストリームの「repeat_first_field」は「1」であるので、トップフィールドT2とボトムフィールドB2とから構成されるフレームの符号化ストリームをデコードする時には、リピートフィールドB2'を生成し、トップフィールドT4とボトムフィールドB4とから構成されるフレームの符号化ストリームをデコードする時には、リピートフィールドB4'を生成する処理を行なう。

【0029】さらに、MPEG規格においては、フレームを構成する2つのフィールドのうち、最初のフィールドがトップフィールドかボトムフィールドかを示す「top_field_first」というフラグを、符号化ストリーム中に記述することを定義している。具体的には、「top_field_first」が「1」である場合には、トップフィールドがボトムフィールドよりも時間的に早いフレーム構造であることを示し、「top_field_first」が「0」の場合には、トップフィールドがボトムフィールドよりも時間的に早いフレーム構造であることを示す。

【0030】図5に示した例では、トップフィールドT1とボトムフィールドB1から構成されるフレームの符号化ストリームの「top_field_first」は「0」であって、トップフィールドT2とボトムフィールドB2から構成されるフレームの符号化ストリームの「top_field_first」は「1」であって、トップフィールドT3とボトムフィールドB3から構成されるフレームの符号化ストリームの「top_field_first」は「0」であって、トップフィールドT4とボトムフィールドB4から構成されるフレームの符号化ストリームの「top_field_first」は「1」である。

【0031】次に、図6を参照して、符号化ストリームをスプライシング処理したときの「top_field_first」及び「repeat_first_field」等のMPEG規格において定義されているフラグに関して発生する問題点について説明する。

【0032】図6(A)は、本局において制作されたオリジナルテレビ番組を符号化したオリジナルストリームSTOLDのフレーム構造を示している図であって、図5(B)は、地方局において制作された地方向けの商業チャネルC1'を符号化した差し替えストリームSTNEWのフレーム構造を示している図であって、図5(C)は、スプライシング処理されたスプライズストリームSTSPLのフレーム構造を示している図である。

【0033】オリジナルストリームSTOLDにおけるプログラム1及びプログラム2は、共に2:3プルダウン処理された符号化ストリームであって、本編の商業チャネルCM1の各フレームは、「top_field_first」が

「0」となっているフレーム構造の符号化ストリームである。また、図6(B)に示された、地方商業CM1'は、オリジナルテレビ番組の商業CM1の部分に差し替えられる符号化ストリームであって、「top_field_first」が「1」となっているフレーム構造の符号化ストリームである。図6(C)に示されたスプライドストリームSTSPLは、プログラム1で示されるオリジナルストリームSTOLDの後に、差し替えストリームSTNEWをスプライスし、さらに、差し替えストリームSTNEWの後に、プログラム2で示されるオリジナルストリームSTOLDをスプライスすることによって生成されたストリームである。つまり、スプライドストリームSTSPLは、オリジナルストリームSTOLDの本編商業CM1の替わりに地方商業CM1'を挿入したストリームである。

【0034】図6に示した本局において制作された商業CM1の各フレームは、top_field_firstが「0」のフレーム構造の符号化ストリームであって、地方局において制作された商業CM1'は、top_field_firstが「1」のフレーム構造の符号化ストリームであることを示している。

【0035】図6(A)及び図6(B)に示したように、商業CM1のフレーム構造と、商業CM1'に対して差し替えられる差し替え商業CM1'のフレーム構造とが異なる場合に、オリジナルストリームSPOLDのスプライスポイントSP1で、プログラム1のストリームの後に商業CM1'のストリームをスプライスすると、スプライドストリームSTSPLにおいてフィールドのギャップが生じてしまう。フィールドのギャップとは、図6(C)に示されたように、スプライスポイントSP1におけるボトムフィールドB6が、スプライドストリームSTSPLから脱落し、トップフィールドとボトムフィールドの繰り返しパターンが不連続になっていることを意味している。

【0036】このようにフィールドのギャップが生じてフィールドパターンが不連続になっている符号化ストリームは、MPEG規格違反の符号化ストリームであって、通常のMPEGデコーダでは正常に復号化できない。

【0037】また、図6(A)及び図6(B)に示したように、オリジナルストリームSPOLDのスプライスポイントSP2で、商業CM1'のストリームの後にプログラム2のストリームをスプライスすると、スプライドストリームSTSPLにおいてフィールドの重複が生じてしまう。このフィールドの重複とは、図6(C)に示されたように、スプライスポイントSP2におけるボトムフィールドb12及びボトムフィールドB12が同じ表示時間に存在することになってしまうことを意味している。

【0038】このようにフィールドの重複が生じてフィ

ールドパターンが不連続になっている符号化ストリームは、MPEG規格違反の符号化ストリームであって、通常のMPEGデコーダでは正常に復号化できない。

【0039】つまり、単純にスプライシング処理を行なうと、フィールドパターン/フレームパターンが不連続になってしまい、MPEG規格に準じたスプライドストリームを生成することができなかった。

【0040】

【発明が解決しようとする課題】本発明の目的は、スプライシングされたストリームのVBVバッファのデータ占有量の軌跡が連続であって、且つ、VBVバッファが破綻しないシームレスなスプライシング処理を実現するための符号化ストリームスプライシング装置を提供することにある。さらには、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならないシームレスなスプライシング処理を実現するための符号化ストリームスプライシング装置を提供することにある。

【0041】

【課題を解決するための手段】請求項1に記載の符号化ストリームスプライシング装置は、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第1の符号化ストリームの符号化パラメータに基づいて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリームとがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシング手段とを備えたことを特徴とする。

【0042】請求項11に記載の符号化ストリームスプライシング方法は、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第1の符号化ストリームの符号化パラメータに基づいて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリームとがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシングステップとを備えたことを特徴とする。

【0043】請求項21に記載の符号化ストリーム生成装置は、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第1の符号化ストリームの符号化パラメータに基づいて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリーム

とがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシング手段とを備えたことを特徴とする。

【0044】請求項22に記載の符号化ストリーム生成方法は、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第1の符号化ストリームの符号化パラメータに基づいて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリームとがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシングステップとを備えたことを特徴とする。

【0045】請求項23に記載の情報処理装置は、データを受信する受信手段と、受信手段で受信したデータの一部を、他のデータと入れ替える入れ替え手段と、他のデータを符号化する符号化手段と、受信手段で受信したデータを、他のデータと整合が取れるように変更する変更手段とを備えることを特徴とする。

【0046】請求項27に記載の情報処理方法は、データを受信する受信ステップと、受信ステップで受信したデータの一部を、他のデータと入れ替える入れ替えステップと、他のデータを符号化する符号化ステップと、受信ステップで受信したデータを、他のデータと整合が取れるように変更する変更ステップとを含むことを特徴とする。

【0047】請求項1の符号化ストリームスプライシング装置及び請求項21の符号化ストリーム生成装置においては、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第1の符号化ストリームの符号化パラメータに基づいて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリームとがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシング手段とを備えることによって、スプライシングされたストリームのVBVバッファのデータ占有量の軌跡が連続であって、且つ、VBVバッファが破綻しないシームレスなスプライシング処理を実現することができる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【0048】また、請求項11の符号化ストリームスプライシング方法及び請求項22の符号化ストリーム生成方法においては、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第1の符号化ストリームの符号化パラメータに基づいて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリームとがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシングステップとを行なうことによって、スプライシングされたストリームのVBVバッファのデータ占有量の軌跡が連続であって、且つ、VBVバッファが破綻しないシームレスなスプライシング処理を実現することができる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【0049】請求項23の情報処理装置及び請求項27の情報処理方法においては、データが受信され、受信されたデータの一部が、他のデータと入れ替えられ、他のデータが符号化され、受信したデータが、他のデータと整合が取れるように変更される。

【0050】

【発明の実施の形態】図7は、本発明の実施の形態に係る符号化ストリームスプライシング装置を含むデジタル放送システムを構成を示す図である。

【0051】図7に示されているように、一般的には、デジタル放送システムは主局(Keystation)30と、この主局系列の地方局40とから構成されている。

【0052】主局30は、系列の地方局に対して共通のテレビプログラムを制作及び伝送するための放送局であって、放送システムコントローラ31と、素材サーバ32と、CMサーバ33と、マトリックススイッチャブロック34と、MPEGエンコーダブロック35と、マルチプレクサ36と、変調回路37とから構成されている。

【0053】放送システムコントローラ31は、素材サーバ32、CMサーバ33、マトリックススイッチャブロック34、MPEGエンコーダブロック35、マルチプレクサ36、及び変調回路37等の放送局に設けられている全ての装置及び回路を統合的に管理及びコントロールするシステムである。この放送システムコントローラ31は、番組供給会社から供給された番組素材及びプロモーション素材や自局で作成した番組素材やCM素材などのあらゆる素材の放映時間を管理するための番組編成表が登録されており、放送番組編成システム1は、こ

の番組編成表に従って、上述した各装置及び回路を制御する。この番組編成表は、例えば、1時間単位又は1日単位の放送番組スケジュールが記録されているイベント情報ファイル、及び15秒単位の放送番組のタイムスケジュールが記録されている運行情報ファイル等から構成されている。

【0054】素材サーバ32は、テレビ番組の本編として放送される映画番組、スポーツ番組、娯楽番組、ニュース番組等のビデオデータ及びオーディオデータを格納すると共に、放送システムコントローラ31によって指定されたプログラムを、番組編成表のタイムスケジュールに従ったタイミングで出力するためのサーバである。また、この映画番組とは、先に説明したように、24Hzのフレーム周波数を有するフィルム素材から2:3プルダウン処理することによって30Hzのフレーム周波数を有するテレビジョン信号に変換されたビデオデータである。この素材サーバ32から本編のプログラムとして出力されたビデオデータ及びオーディオデータはマトリックススイッチャブロック34に供給される。例えば、図2に示した例では、プログラム1、プログラム2、プログラム3及びプログラム4等がこの素材サーバ32に記録されている。尚、この素材サーバ32に記憶されているビデオデータ及びオーディオデータは、圧縮符号化されていないベースバンドのビデオデータ及びオーディオデータである。

【0055】CMサーバ33は、素材サーバ32から再生された本編のプログラム間に挿入されるコマーシャルを格納すると共に、放送システムコントローラ31によって指定されたコマーシャルを、番組編成表のタイムスケジュールに従ったタイミングで出力するためのサーバである。このCMサーバ33からコマーシャルとして出力されたビデオデータ及びオーディオデータはマトリックススイッチャ34に供給される。例えば、図2に示した例では、コマーシャルCM1、コマーシャルCM2、及びコマーシャルCM3等が、このCMサーバ33に記録されている。尚、このCMサーバ33に記憶されているビデオデータ及びオーディオデータは、圧縮符号化されていないベースバンドのビデオデータ及びオーディオデータである。

【0056】マトリックススイッチャブロック34は、スポーツ中継やニュース番組等のライブプログラム、素材サーバ32から出力された本編プログラム、及びCMサーバ33から出力されたコマーシャルプログラムをそれぞれルーティングするマトリックス回路を有している。さらに、マトリックススイッチャブロック34は、放送システムコントローラによって決定された番組編成表のタイムスケジュールに従ったタイミングで、素材サーバ32から供給された本編プログラムとCMサーバ33から供給されたコマーシャルプログラムを接続してスイッチングするスイッチング回路を有している。このス

イッチング回路によって、本編プログラムとCMプログラムをスイッチングすることによって、例えば図2に示した伝送プログラムPGOLDを生成することができる。

【0057】MPEGエンコーダブロック35は、マトリックススイッチャブロックから出力されたベースバンドのビデオデータ及びオーディオデータをMPEG2規格に基いて符号化するためのブロックであって、複数のビデオエンコーダ及びオーディオエンコーダを有している。

【0058】マルチプレクサ36は、MPEGエンコーダブロック35から出力された9チャンネルのトランスポートストリームを多重化して、1つの多重化トランスポートストリームを生成する。従って、出力される多重化トランスポートストリーム中には、1から9チャンネルまでの符号化ビデオエレメンタリーストリームを含んだトランスポートストリームパケットと1チャンネルから9チャンネルまでの符号化オーディオエレメンタリーストリームを含んだトランスポートストリームパケットとが混在しているストリームである。

【0059】変調回路37は、トランスポートストリームをQPSK変調して、その変調されたデータを伝送路を介して、地方局40及び家庭61に伝送する。

【0060】次に、地方局40の全体構成に関して、図7を参照して説明する。

【0061】地方局40は、本局から送られてきた共通のテレビプログラムを地方向けに編集し、地方向けに編集されたテレビ番組を各家庭に放映するための放送局であって、放送システムコントローラ41と、復調回路42と、デマルチプレクサ43と、ストリーム変換回路44と、素材サーバ46と、CMサーバ47と、エンコーダブロック48と、ストリームサーバ49と、ストリームスプライサ50と、ストリーム変換回路51と、マルチプレクサ52と、変調回路53とを備えている。

【0062】放送システムコントローラ41は、本局30の放送システムコントローラ31と同じように、復調回路42、デマルチプレクサ43、ストリーム変換回路44、素材サーバ46、CMサーバ47、エンコーダブロック48、ストリームサーバ49、ストリームスプライサ50、ストリーム変換回路51、マルチプレクサ52、変調回路53等の地方局に設けられている全ての装置及び回路を統合的に管理及びコントロールするシステムである。この放送システムコントローラ41は、本局30における放送システムコントローラ31と同様に、本局30から供給された伝送プログラムに対して、自局で作成した番組やCMなどを挿入した編集テレビ番組の放映時間を管理するための番組編成表が登録されており、この番組編成表に従って、上述した各装置及び回路を制御する。

【0063】復調回路42は、伝送路を介して本局30から伝送された伝送プログラムをQPSK復調すること

によって、トランスポートストリームを生成する。

【0064】デマルチプレクサ43は、復調回路42から出力されたトランスポートストリームをデマルチプレックスして9チャンネルのトランスポートストリームを生成し、それぞれのチャンネルのトランスポートストリームをストリーム変換回路44に出力する。つまり、このデマルチプレクサ43は、本局30のマルチプレクサ36とは逆の処理を行なう。

【0065】ストリーム変換回路44は、デマルチプレクサ43から供給されたトランスポートストリームをエレメンタリーストリームの形態に変換するための回路である。

【0066】素材サーバ46は、地方向けのテレビ番組として放送される娯楽番組、ニュース番組等のビデオデータ及びオーディオデータを格納しているサーバである。CMサーバ47は、本局30から供給された本編プログラム間に挿入される地方向けコマーシャルのビデオデータ及びオーディオデータを格納するためのサーバである。この素材サーバ46及びCMサーバ47に記憶されているビデオデータ及びオーディオデータは、圧縮符号化されていないベースバンドのビデオデータ及びオーディオデータである。

【0067】エンコーダブロック48は、素材サーバ46及びCMサーバ47から供給された複数チャンネルのビデオデータ及び複数チャンネルのオーディオデータを符号化するためのブロックであって、複数チャンネルに対応した複数のビデオエンコーダ及び複数のオーディオエンコーダを備えている。このエンコーダブロック48と本局30のMPEGエンコーダブロック35との違いは、本局30のMPEGエンコーダブロック35はトランスポートストリームを出力するのに対して、この地方局40のエンコーダブロック48は、エレメンタリーストリーム出力する点で異なっているが、このエンコーダブロック48の実質的な機能及び処理は本局30のMPEGエンコーダブロック35と全く同じである。エンコーダブロック48から出力された複数チャンネルのエレメンタリーストリームのうち3チャンネル分のエレメンタリーストリームが、ストリームサーバ49に供給され、残りのチャンネルのエレメンタリーストリームは、ストリームスプライサ50に供給される。

【0068】ストリームサーバ49は、エンコーダブロックから供給された3チャンネル分のエレメンタリーストリームを受取り、ストリームの状態でランダムアクセス可能な記録媒体にストリームの状態で記録すると共に、放送システムコントローラ41からの制御にしたがって、そのエレメンタリーストリームをランダムアクセス可能な記録媒体から再生する。

【0069】ストリームスプライサ50は、エンコーダブロック48及びストリームサーバ49から供給された複数のエレメンタリーストリームをルーティングして所定の出力

ラインに出力すると共に、本局30から供給されたエレメンタリーストリームと、地方局40において生成されたエレメンタリーストリームとをストリームレベルでスプライシングするためのブロックである。このストリームスプライサ50における処理は、詳しくは後述する。

【0070】ストリーム変換回路51は、ストリームスプライサ50からスプライズドストリームとして出力されたエレメンタリーストリームを受けとり、このエレメンタリーストリームをトランスポートストリームに変換する回路である。

【0071】マルチプレクサ52は、本局30のマルチプレクサ36と同じように、ストリーム変換回路から出力された9チャンネルのトランスポートストリームを多重化して、1つの多重化トランスポートストリームを生成する。

【0072】変調回路53は、トランスポートストリームをQPSK変調して、その変調されたデータを伝送路を介して、各家庭62に配信する。

【0073】図8は、本局30のMPEGエンコーダブロック35及び地方局40のエンコーダブロック48の構成を詳細に説明するためのブロック図である。本局30のMPEGエンコーダブロック35と地方局40のエンコーダブロック48とは実質的に同じ構成であるので、本局30のMPEGエンコーダブロック35を例にあげて、その構成及び処理を説明する。

【0074】MPEGエンコーダブロック35は、このMPEGエンコーダブロック35の全ての回路を集中的に制御するためのエンコーダコントローラ350と、供給された複数チャンネルのビデオデータをエンコードするための複数のMPEGビデオエンコーダ351-1V~351-9Vと、ビデオデータにそれぞれ対応する複数のオーディオデータをMPEG2規格に基いて符号化するためのMPEGオーディオエンコーダ351-1A~351-9Aとを有している。

【0075】さらに、MPEGエンコーダブロック35は、各ビデオエンコーダ351-1V~351-9Vからそれぞれ出力された符号化エレメンタリーストリーム(ES)をトランスポートストリームに変換するストリーム変換回路352-1V~352-9Vと、各オーディオエンコーダ351-1A~351-9Aからそれぞれ出力された符号化エレメンタリーストリーム(ES)をトランスポートストリームに変換するストリーム変換回路352-1A~352-9Aと、第1チャンネル(1ch)のビデオエレメンタリーストリームを含んだトランスポートストリームと、第1チャンネル(1ch)のオーディオエレメンタリーストリームを含んだトランスポートストリームとをトランスポートストリームパケット単位で多重化するマルチプレクサ353-1と、第2チャンネル(2ch)のビデオエレメンタリーストリームを含んだトランスポートストリームと、第2

チャンネル(2ch)のオーディオエレメンタリーストリームを含んだトランスポートストリームとをトランスポートストリームパケット単位で多重化するマルチプレクサ353-2と、.....、第9チャンネル(9ch)のビデオエレメンタリーストリームを含んだトランスポートストリームと、第9チャンネル(9ch)のオーディオエレメンタリーストリームを含んだトランスポートストリームとをトランスポートストリームパケット単位で多重化するマルチプレクサ353-9とを備えている。

【0076】図8に示したMPEGエンコードブロック35は、9チャンネルの伝送プログラムをエンコードする構成となっているが、9チャンネルに限らず何チャンネルであっても良いことは言うまでもない。

【0077】図8に示したMPEGエンコードブロック35は、各チャンネルの伝送プログラムの伝送レートを、符号化するビデオデータの絵柄に応じてダイナミックに変化させる統計多重という制御を行なう。この統計多重という手法は、あるチャンネルの伝送プログラムのピクチャの絵柄が比較的簡単であって、このピクチャを符号化するためにそれほど多くのビットを必要としない場合であって、一方、その他のプログラムのピクチャの絵柄が比較的難しく、このピクチャを符号化するために多くのビットを必要とするような場合には、あるチャンネルのピクチャを符号化するためのビットを、その他のチャンネルのピクチャを符号化するビットに割当てることによって、伝送路の伝送レートを効率良くしようすることができる方法である。以下に、このようにダイナミッ

$$R_i = (D_i / \sum D_k) \times \text{Total_Rate} \quad (1)$$

【0080】式(1)において、「 R_i 」は「 i 」チャンネルの伝送プログラムのターゲットビットレート、「 D_i 」は「 i 」チャンネルの伝送プログラムのピクチャを符号化するためのディフィカルティデータ、「 \sum 」は $k=1\sim 9$ チャンネルのディフィカルティデータの総和を意味する。

【0081】エンコードコントローラ350は、式(1)に基いて演算したターゲットビットレート $R_1\sim R_9$ を、それぞれ対応するビデオエンコード351-1V \sim 351-9Vに供給する。このターゲットビットレート $R_1\sim R_9$ を演算する単位は、各ピクチャ毎であっても良いし、またGOP単位であっても良い。

【0082】ビデオエンコード351-1V \sim 351-9Vは、エンコードコントローラ350から供給されたターゲットビットレート $R_1\sim R_9$ をそれぞれ受取り、このターゲットビットレート $R_1\sim R_9$ に対応するように符号化処理を行なう。このように符号化するピクチャの符号化難易度を示すディフィカルティデータに基いて、各ビデオエンコードから出力される符号化ストリームのビットレートをダイナミックに変更することによって、符号化すべきピクチャの符号化難易度に対して最適

クに各ビデオエンコードの符号化レートを変化させる方法について簡単に説明する。

【0078】各ビデオエンコード351-1V \sim 351-9Vは、まず、符号化処理の前に行われる動き補償の結果得られた動き補償残差やイントラAC等の統計量から、符号化対象となっているピクチャを符号化するためにどの程度のビット量が必要かを示すディフィカルティデータ(Difficulty Data) $D_1\sim D_9$ を生成する。このディフィカルティデータとは、符号化難易度を示す情報であって、ディフィカルティが大きいということは符号化対象となっているピクチャの絵柄が複雑であることを表わし、ディフィカルティが小さいということは符号化対象となっているピクチャの絵柄が簡単であることを表わしている。このディフィカルティデータは、ビデオエンコードにおける符号化処理時に使用される、イントラACや動き補償残差(ME残差)等の統計量に基いて、概算することができる。

【0079】エンコードコントローラ350は、各ビデオエンコード351-1V \sim 351-9Vからそれぞれ出力されたディフィカルティデータ $D_1\sim D_9$ を受取り、それらのディフィカルティデータ $D_1\sim D_9$ に基いて、各ビデオエンコード351-1V \sim 351-9Vに対するターゲットビットレート $R_1\sim R_9$ をそれぞれ演算する。具体的には、以下の式(1)に示すように、エンコードコントローラ350は、伝送路のトータル伝送レート Total_Rate を、ディフィカルティデータ $D_1\sim D_9$ を用いて比例配分することによって、ターゲットビットレート $R_1\sim R_9$ を求めることができる。

な量のビットを割り当てることができ、さらに、各ビデオエンコードからそれぞれ出力されるビットレートの総量が、伝送路のトータル伝送レート Total_Rate をオーバーフローすることがない。

【0083】ストリーム変換回路352-1V \sim 352-9V及び、ストリーム変換回路352-1A \sim 352-9Aは、共に、エレメンタリーストリームをトランスポートストリームに変換するための回路である。

【0084】図9を参照して、ビデオエンコード351-1Vにおいて、供給されたソースビデオデータを符号化してビデオエレメンタリーストリームを生成し、そのビデオエレメンタリーストリームを、ストリーム変換回路352-1Vにおいてトランスポートストリームに変換する例を挙げて、ビデオエレメンタリーストリームからトランスポートストリームを生成する過程について説明する。

【0085】図9(A)は、ビデオエンコード351-1Vに供給されるソースビデオデータを示し、図9(B)は、ビデオエンコード351-1Vから出力されたビデオエレメンタリーストリーム(ES)を示し、図9(C)は、パケッライズドエレメンタリーストリーム

(PES)、図9(D)は、トランスポートストリーム(TS)を示している。

【0086】図9(B)に示したストリームV1、V2、V3及びV4のように、MPEG2規格において符号化されたエレメンタリーストリームのデータ量は、ビデオフレームのピクチャタイプ(Iピクチャ、Pピクチャ又はBピクチャ)及び動き補償の有無に応じて異なってくる。図9(C)に示したパケッタイズドエレメンタリーストリーム(PES)は、その複数のエレメンタリーストリームをパケット化し、その先頭にPESヘッダを付加することによって生成される。例えば、このPESヘッダには、PESパケットの開始を示す24[bit]のパケット開始コードと、PESパケットの実データ部分に収容されるストリームデータの種別(例えばビデオや音声等の種別)を示す8[bit]のストリームIDと、以降に続くデータの長さを示す16[bit]のパケット長と、値「10」を示すコードデータと、各種フラグ情報が格納されるフラグ制御部と、コンディショナル・コーディング部のデータの長さを示す8[bit]のPESヘッダ長と、PTS(Presentation Time Stamp)と呼ばれる再生出力の時間情報やDTS(Decoding Time Stamp)と呼ばれる復号時の時刻管理情報、或いはデータ量調整のためのスタッフィングバイト等が格納される可変長のコンディショナル・コーディング部とによって構成される。

【0087】トランスポートストリーム(TS)は、4バイトのTSヘッダと184バイトの実データが記録されるペイロード部とから成るトランスポートストリームパケットのデータ列である。このトランスポートストリームパケット(TSパケット)を生成するためには、まず、PESパケットのデータストリームを184バイト毎に分解し、その184バイトの実データをTSパケットのペイロード部に挿入し、その184バイトのペイロードのデータに4バイトのTSヘッダを付加することによってトランスポートストリームパケットが生成される。

【0088】次に、図10から図17を参照してエレメンタリーストリームのシンタックス及び構造について説明すると共に、図18から図19を参照してトランスポートストリームのシンタックス及び構造について詳しく説明する。

【0089】図10は、MPEGのビデオエレメンタリーストリームのシンタックスを表わした図である。ビデオエンコードブロック35内の各ビデオエンコード351-1V~3519Vは、この図10に示されたシンタックスに従った符号化エレメンタリーストリームを生成する。以下に説明するシンタックスにおいて、関数や条件文は細活字で表わされ、データエレメントは、太活字で表されている。データ項目は、その名称、ビット長及びそのタイプ・伝送順序を示すニーモニック(Mnemoni

c)で記述されている。

【0090】まず、この図10に示されているシンタックスにおいて使用されている関数について説明する。実際には、この図10に示されているシンタックスは、ビデオデコード側において伝送された符号化ストリームから所定の意味のあるデータを抽出するために使用されるシンタックスである。ビデオエンコード側において使用されるシンタックスは、図10に示されたシンタックスからif分やwhile文等の条件文を省略したシンタックスである。

【0091】video_sequesce()において最初に記述されているnext_start_code()関数は、ビットストリーム中に記述されているスタートコードを探すための関数である。この図6に示されたシンタックスに従って生成された符号化ストリームには、まず最初に、sequence_header()関数とsequence_extension()関数によって定義されたデータエレメントが記述されている。このsequence_header()関数は、MPEGビットストリームのシーケンスレイヤのヘッダデータを定義するための関数であって、sequence_extension()関数は、MPEGビットストリームのシーケンスレイヤの拡張データを定義するための関数である。

【0092】sequence_extension()関数の次に配置されているdo{}while構文は、while文によって定義されている条件が真である間、do文の{}内の関数に基いて記述されたデータエレメントが符号化データストリーム中に記述されていることを示す構文である。このwhile文に使用されているnextbits()関数は、ビットストリーム中に記述されているビット又はビット列と、参照されるデータエレメントとを比較するための関数である。この図6に示されたシンタックスの例では、nextbits()関数は、ビットストリーム中のビット列とビデオシーケンスの終わりを示すsequence_end_codeとを比較し、ビットストリーム中のビット列とsequence_end_codeとが一致しないときに、このwhile文の条件が真となる。従って、sequence_extension()関数の次に配置されているdo{}while構文は、ビットストリーム中に、ビデオシーケンスの終わりを示すsequence_end_codeが現れない間、do文中の関数によって定義されたデータエレメントが符号化ビットストリーム中に記述されていることを示している。

【0093】符号化ビットストリームにおいて、sequence_extension()関数によって定義された各データエレメントの次には、extension_and_user_data(0)関数によって定義されたデータエレメントが記述されている。このextension_and_user_data(0)関数は、MPEGビットストリームのシーケンスレイヤにおける拡張データとユーザデータを定義するための関数である。

【0094】このextension_and_user_data(0)関数の次に配置されているdo{}while構文は、while文によって

定義されている条件が真である間、do文の{}内の関数に基いて記述されたデータエレメントが、ビットストリーム中に記述されていることを示す関数である。このwhile文において使用されているnextbits()関数は、ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとの一致を判断するための関数である。ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとが一致する場合には、while文によって定義された条件が真となる。よって、このdo{}while構文は、符号化ビットストリーム中において、picture_start_code又はgroup_start_codeが現れた場合には、そのスタートコードの次に、do文中の関数によって定義されたデータエレメントのコードが記述されていることを示している。

【0095】このdo文の最初に記述されているif文は、符号化ビットストリーム中にgroup_start_codeが現れた場合、という条件を示している。このif文による条件は真である場合には、符号化ビットストリーム中には、このgroup_start_codeの次にgroup_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されているデータエレメントが順に記述されている。

【0096】このgroup_of_picture_header()関数は、MPEG符号化ビットストリームのGOPレイヤのヘッダデータを定義するための関数であって、extension_and_user_data(1)関数は、MPEG符号化ビットストリームのGOPレイヤの拡張データ及びユーザデータを定義するための関数である。

【0097】さらに、この符号化ビットストリームにおいて、group_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されているデータエレメントの次に、picture_header()関数とpicture_coding_extension()関数によって定義されたデータエレメントが記述されている。もちろん、先に説明したif文の条件が真とならない場合には、group_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されているデータエレメントは記述されていないので、extension_and_user_data(0)関数によって定義されているデータエレメントの次に、picture_header()関数、picture_coding_extension()関数及びextension_and_user_data(2)関数によって定義されたデータエレメントが記述されている。

【0098】このpicture_header()関数は、MPEG符号化ビットストリームのピクチャレイヤのヘッダデータを定義するための関数であって、picture_coding_extension()関数は、MPEG符号化ビットストリームのピクチャレイヤの第1の拡張データを定義するための関数である。extension_and_user_data(2)関数は、MPEG符号化ビットストリームのピクチャレイヤの拡張データ及びユーザデータを定義するための関数である。このex

tension_and_user_data(2)関数によって定義されるユーザデータは、ピクチャレイヤに記述されているデータであって、各ピクチャ毎に記述することのできるデータであるので、本発明においては、このextension_and_user_data(2)関数によって定義されるユーザデータとして、タイムコード情報を記述するようにしている。

【0099】符号化ビットストリームにおいて、ピクチャレイヤのユーザデータの次には、picture_data()関数によって定義されるデータエレメントが記述されている。このpicture_data()関数は、スライスレイヤ及びマクロブロックレイヤに関するデータエレメントを記述するための関数である。

【0100】このpicture_data()関数の次に記述されているwhile文は、このwhile文によって定義されている条件が真である間、次のif文の条件判断を行うための関数である。このwhile文において使用されているnextbits()関数は、符号化ビットストリーム中に、picture_start_code又はgroup_start_codeが記述されているか否かを判断するための関数であって、ビットストリーム中にpicture_start_code又はgroup_start_codeが記述されている場合には、このwhile文によって定義された条件が真となる。

【0101】次のif文は、符号化ビットストリーム中にsequence_end_codeが記述されているか否かを判断するための条件文であって、sequence_end_codeが記述されていないのであれば、sequence_header()関数とsequence_extension()関数とによって定義されたデータエレメントが記述されていることを示している。sequence_end_codeは符号化ビデオストリームのシーケンスの終わりを示すコードであるので、符号化ストリームが終了しない限り、符号化ストリーム中にはsequence_header()関数とsequence_extension()関数とによって定義されたデータエレメントが記述されている。

【0102】このsequence_header()関数とsequence_extension()関数によって記述されたデータエレメントは、ビデオストリームのシーケンスの先頭に記述されたsequence_header()関数とsequence_extension()関数によって記述されたデータエレメントと全く同じである。このように同じデータをストリーム中に記述する理由は、ビットストリーム受信装置側でデータストリームの途中(例えばピクチャレイヤに対応するビットストリーム部分)から受信が開始された場合に、シーケンスレイヤのデータを受信できなくなり、ストリームをデコード出来なくなることを防止するためである。

【0103】この最後のsequence_header()関数とsequence_extension()関数とによって定義されたデータエレメントの次、つまり、データストリームの最後には、シーケンスの終わりを示す32ビットのsequence_end_codeが記述されている。

【0104】以下に、sequence_header()関数、sequenc

e_extension()関数、extension_and_user_data(0)関数、group_of_picture_header()関数及びextension_and_user_data(1)関数について詳細に説明する。

【0105】図11は、sequence_header()関数のシンタックスを説明するための図である。このsequence_header()関数によって定義されたデータエレメントは、sequence_header_code、sequence_header_present_flag、horizontal_size_value、vertical_size_value、aspect_ratio_information、frame_rate_code、bit_rate_value、marker_bit、VBV_buffer_size_value、constrained_parameter_flag、load_intra_quantizer_matrix、intra_quantizer_matrix、load_non_intra_quantizer_matrix、及びnon_intra_quantizer_matrix等である。

【0106】sequence_header_codeは、シーケンスレイヤのスタート同期コードを表すデータである。sequence_header_present_flagは、sequence_header内のデータが有効か無効かを示すデータである。horizontal_size_valueは、画像の水平方向の画素数の下位12ビットから成るデータである。vertical_size_valueは、画像の縦のライン数の下位12ビットからなるデータである。aspect_ratio_informationは、画素のアスペクト比（縦横比）または表示画面アスペクト比を表すデータである。frame_rate_codeは、画像の表示周期を表すデータである。bit_rate_valueは、発生ビット量に対する制限のためのビット・レートの下位18ビット（400bsp単位で切り上げる）データである。marker_bitは、スタートコードエミュレーションを防止するために挿入されるビットデータである。VBV_buffer_size_valueは、発生符号量制御用の仮想バッファ（ビデオバッファバリファイヤー）の大きさを決める値の下位10ビットデータである。constrained_parameter_flagは、各パラメータが制限以内であることを示すデータである。load_intra_quantizer_matrixは、イントラMB用量子化マトリックス・データの存在を示すデータである。intra_quantizer_matrixは、イントラMB用量子化マトリックスの値を示すデータである。load_non_intra_quantizer_matrixは、非イントラMB用量子化マトリックス・データの存在を示すデータである。non_intra_quantizer_matrixは、非イントラMB用量子化マトリックスの値を表すデータである。

【0107】図12はsequence_extension()関数のシンタックスを説明するための図である。このsequence_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbv_buffer_size_extension、low_delay、frame_rate_extension_n、及びframe_rate_extension_d等のデータエレメントである。

【0108】extension_start_codeは、エクステンシ

ョンデータのスタート同期コードを表すデータである。extension_start_code_identifierは、どの拡張データが送られるかを示すデータである。sequence_extension_present_flagは、シーケンスエクステンション内のデータが有効であるか無効であるかを示すデータである。profile_and_level_indicationは、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive_sequenceは、ビデオデータが順次走査であることを示すデータである。chroma_formatは、ビデオデータの色差フォーマットを指定するためのデータである。horizontal_size_extensionは、シーケンスヘッダのhorizontal_size_valueに加える上位2ビットのデータである。vertical_size_extensionは、シーケンスヘッダのvertical_size_valueに加える上位2ビットのデータである。bit_rate_extensionは、シーケンスヘッダのbit_rate_valueに加える上位12ビットのデータである。vbv_buffer_size_extensionは、シーケンスヘッダのvbv_buffer_size_valueに加える上位8ビットのデータである。low_delayは、Bピクチャを含まないことを示すデータである。frame_rate_extension_nは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを得るためのデータである。frame_rate_extension_dは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを得るためのデータである。

【0109】図13は、extension_and_user_data(i)関数のシンタックスを説明するための図である。このextension_and_user_data(i)関数は、「i」が2以外のときは、extension_data()関数によって定義されるデータエレメントは記述せずに、user_data()関数によって定義されるデータエレメントのみを記述する。よって、extension_and_user_data(0)関数は、user_data()関数によって定義されるデータエレメントのみを記述する。

【0110】図14は、group_of_picture_header()関数のシンタックスを説明するための図である。このgroup_of_picture_header()関数によって定義されたデータエレメントは、group_start_code、group_of_picture_header_present_flag、time_code、closed_gop、及びbroken_linkから構成される。

【0111】group_start_codeは、GOPレイヤの開始同期コードを示すデータである。group_of_picture_header_present_flagは、group_of_picture_header内のデータエレメントが有効であるか無効であるかを示すデータである。time_codeは、GOPの先頭ピクチャのシーケンスの先頭からの時間を示すタイムコードである。closed_gopは、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。broken_linkは、編集などのためにGOP内の先頭のBピクチャが正確に再生できないことを示すフラグデータである。

【0112】extension_and_user_data(1)関数は、extension_and_user_data(0)関数と同じように、user_data

()関数によって定義されるデータエレメントのみを記述するための関数である。

【0113】次に、図15から図17を参照して、符号化ストリームのピクチャレイヤに関するデータエレメントを記述するためのpicture_headr()関数、picture_coding_extension()関数、extensions_and_user_data(2)及びpicture_data()について説明する。

【0114】図15はpicture_headr()関数のシンタックスを説明するための図である。このpicture_headr()関数によって定義されたデータエレメントは、picture_start_code、temporal_reference、picture_coding_type、v_bv_delay、full_pel_forward_vector、forward_f_code、full_pel_backward_vector、backward_f_code、extra_bit_picture、及びextra_information_pictureである。

【0115】具体的には、picture_start_codeは、ピクチャレイヤの開始同期コードを表すデータである。temporal_referenceは、ピクチャの表示順を示す番号でGOPの先頭でリセットされるデータである。picture_coding_typeは、ピクチャタイプを示すデータである。

【0116】v_bv_delayは、VBVバッファの初期状態を示すデータであって、各ピクチャ毎に設定されている。送信側システムから受信側システムに伝送された符号化エレメントストリームのピクチャは、受信側システムに設けられたVBVバッファにバッファリングされ、DTS (Decoding Time Stamp) によって指定された時間に、このVBVバッファから引き出され(読み出され)、デコードに供給される。v_bv_delayによって定義される時間は、復号化対象のピクチャがVBVバッファにバッファリングされ始めてから、符号化対象のピクチャがVBVバッファから引き出される時間、つまりDTSによって指定された時間までを意味する。本発明の符号化ストリームスプライシング装置においては、このピクチャヘッダに格納されたv_bv_delayを使用することによって、VBVバッファのデータ占有量不連続にならないシームレスなスプライシングを実現するようにしている。詳しくは後述する。

【0117】full_pel_forward_vectorは、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。forward_f_codeは、順方向動きベクトル探索範囲を表すデータである。full_pel_backward_vectorは、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。backward_f_codeは、逆方向動きベクトル探索範囲を表すデータである。extra_bit_pictureは、後続する追加情報の存在を示すフラグである。このextra_bit_pictureが「1」の場合には、次にextra_information_pictureが存在し、extra_bit_pictureが「0」の場合には、これに続くデータが無いことを示している。extra_information_pictureは、規格において予約された情報である。

【0118】図16は、picture_coding_extension()関数のシンタックスを説明するための図である。このpicture_coding_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、f_code[0][0]、f_code[0][1]、f_code[1][0]、f_code[1][1]、intra_dc_precision、picture_structure、top_field_first、frame_predictive_frame_dct、concealment_motion_vectors、q_scale_type、intra_vlc_format、alternate_scan、repeat_first_field、chroma_420_type、progressive_frame、composite_display_flag、v_axis、field_sequence、sub_carrier、burst_amplitude、及びsub_carrier_phaseから構成される。

【0119】extension_start_codeは、ピクチャレイヤのエクステンションデータのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。intra_dc_precisionは、DC係数の精度を表すデータである。picture_structureは、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。

【0120】top_field_firstは、フレームストラクチャの場合、最初のフィールドがトップフィールドであるのかボトムフィールドであるのかを示すフラグである。frame_predictive_frame_dctは、フレーム・ストラクチャの場合、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。concealment_motion_vectorsは、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。q_scale_typeは、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_formatは、イントラマクロブロックに、別の2次元VLCを使うかどうかを示すデータである。alternate_scanは、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。

【0121】repeat_first_fieldは、復号化時にリピートフィールドを生成するか否かを示すフラグであって、復号化時の処理において、repeat_first_fieldが「1」の場合にはリピートフィールドを生成し、repeat_first_fieldが「0」の場合にはリピートフィールドを生成しないという処理を行なう。chroma_420_typeは、信号フォーマットが4:2:0の場合、次のprogressive

e_frameと同じ値、そうでない場合は0を表すデータである。progressive_frameは、このピクチャが、順次走査できているかどうかを示すデータである。composite_display_flagは、ソース信号がコンポジット信号であったかどうかを示すデータである。v_axisは、ソース信号が、PALの場合に使われるデータである。field_sequenceは、ソース信号が、PALの場合に使われるデータである。sub_carrierは、ソース信号が、PALの場合に使われるデータである。burst_amplitudeは、ソース信号が、PALの場合に使われるデータである。sub_carrier_phaseは、ソース信号が、PALの場合に使われるデータである。

【0122】extension_and_user_data(2)関数は、図13に示したように、符号化ビットストリーム中にエクステンションスタートコードextension_start_codeが存在する場合には、extension_data()関数によって定義されるデータエレメントが記述されている。但し、ビットストリーム中にエクステンションスタートコードが存在しない場合にはextension_data()関数によって定義されるデータエレメントはビットストリーム中には記述されていない。このextension_data()関数によって定義されているデータエレメントの次には、ビットストリーム中にユーザデータスタートコードuser_data_start_codeが存在する場合には、user_data()関数によって定義されるデータエレメントが記述されている。

【0123】図17は、picture_data()関数のシンタックスを説明するための図である。このpicture_data()関数によって定義されるデータエレメントは、slice()関数によって定義されるデータエレメントである。但し、ビットストリーム中に、slice()関数のスタートコードを示すslice_start_codeが存在しない場合には、このslice()関数によって定義されるデータエレメントはビットストリーム中に記述されていない。

【0124】slice()関数は、スライスレイヤに関するデータエレメントを記述するための関数であって、具体的には、slice_start_code、slice_quantiser_scale_code、intra_slice_flag、intra_slice、reserved_bits、extra_bit_slice、extra_information_slice、及びextra_bit_slice等のデータエレメントと、macroblock()関数によって定義されるデータエレメントを記述するための関数である。

【0125】slice_start_codeは、slice()関数によって定義されるデータエレメントのスタートを示すスタートコードである。slice_quantiser_scale_codeは、このスライスレイヤに存在するマクロブロックに対して設定された量子化ステップサイズを示すデータである。しかし、各マクロブロック毎に、quantiser_scale_codeが設定されている場合には、各マクロブロックに対して設定されたmacroblock_quantiser_scale_codeのデータが優先して使用される。intra_slice_flagは、ビットストリ

ーム中にintra_slice及びreserved_bitsが存在するか否かを示すフラグである。intra_sliceは、スライスレイヤ中にノンイントラマクロブロックが存在するか否かを示すデータである。スライスレイヤにおけるマクロブロックのいずれかがノンイントラマクロブロックである場合には、intra_sliceは「0」となり、スライスレイヤにおけるマクロブロックの全てがノンイントラマクロブロックである場合には、intra_sliceは「1」となる。reserved_bitsは、7ビットのデータであって「0」の値を取る。extra_bit_sliceは、符号化ストリームとして追加の情報が存在することを示すフラグであって、次にextra_information_sliceが存在する場合には「1」に設定される。追加の情報が存在しない場合には「0」に設定される。

【0126】macroblock()関数は、マクロブロックレイヤに関するデータエレメントを記述するための関数であって、具体的には、macroblock_escape、macroblock_address_increment、及びmacroblock_quantiser_scale_code等のデータエレメントと、macroblock_modes()関数、及びmacroblock_vectors(s)関数によって定義されたデータエレメントを記述するための関数である。

【0127】macroblock_escapeは、参照マクロブロックと前のマクロブロックとの水平方向の差が34以上であるか否かを示す固定ビット列である。参照マクロブロックと前のマクロブロックとの水平方向の差が34以上の場合には、macroblock_address_incrementの値に33をプラスする。macroblock_address_incrementは、参照マクロブロックと前のマクロブロックとの水平方向の差を示すデータである。もし、このmacroblock_address_incrementの前にmacroblock_escapeが1つ存在するのであれば、このmacroblock_address_incrementの値に33をプラスした値が、実際の参照マクロブロックと前のマクロブロックとの水平方向の差分を示すデータとなる。macroblock_quantiser_scale_codeは、各マクロブロック毎に設定された量子化ステップサイズである。各スライスレイヤには、スライスレイヤの量子化ステップサイズを示すslice_quantiser_scale_codeが設定されているが、参照マクロブロックに対してmacroblock_quantiser_scale_codeが設定されている場合には、この量子化ステップサイズを選択する。

【0128】次に、図18及び図19を参照して、トランスポートストリームパケットの構造及びトランスポートストリームパケットのシンタックスについて詳しく説明する。

【0129】トランスポートストリームパケットは、4バイトのヘッダと、各種のデータ及びデータエレメントを格納するための184バイトのペーロード部とから構成されている。

【0130】トランスポートストリームパケットのヘッダ部は、sync_byte、transport_error_indicator、payl

oad_unit_start_indicator、transport_priority、PID、transport_scrambling_control、adaptation_field_control、continuity_counter、及びadaptation_field等の各種フィールドから構成されている。

【0131】sync_byteとは、ビットストリーム中から同期パターンを検出するための固定の8ビットのフィールドである。値は‘01000111’ (0x47) の固定値で定義され、このストリーム中のこのビットパターンを検出することによって、同期を検出することができる。

【0132】transport_error_indicatorは、1ビットのフラグである。「1」に設定されると、少なくとも1ビットの訂正できないビットエラーがトランスポートストリームパケットに存在することを示す。

【0133】payload_unit_start_indicatorは、1ビットのフラグである。ビデオ／オーディオデータ等のエレメンタリーデータまたはプログラム仕様情報 (PSI) を伝送するトランスポートストリームパケットに対して規範的な意味を有するデータである。トランスポートストリームパケットのペイロードがエレメンタリーデータを含む場合、payload_unit_start_indicatorは、次の意味を有する。payload_unit_start_indicatorが「1」の場合には、このトランスポートストリームパケットのペイロードの最初に、エレメンタリーデータが挿入されていることを示し、payload_unit_start_indicatorが「0」の場合には、このトランスポートストリームパケットのペイロードの最初に、エレメンタリーデータが挿入されていないことを示す。もし、payload_unit_start_indicatorが「1」にセットされると、ただ一つの PESパケットが任意のトランスポートストリームパケットで開始することを示す。一方、トランスポートストリームパケットのペイロードがPSIデータを含む場合、payload_unit_start_indicatorは、次の意味を有する。もし、トランスポートパケットがPSIセクションの第1バイトを伝送する場合、payload_unit_start_indicatorは「1」となる。もし、トランスポートストリームパケットがPSIセクションの第1バイトを伝送していない場合、payload_unit_start_indicatorは「0」となる。尚、トランスポートストリームパケットがヌルパケットの場合にも、payload_unit_start_indicatorは「0」となる。

【0134】transport_priorityは、トランスポートパケットの優先度を示す1ビットの識別子である。このtransport_priorityが「1」に設定されると、このトランスポートパケットは、同一のパケット識別子PIDをもつパケットであって、このtransport_priorityが「1」でないパケットより優先度が高いことを示している。例えば、このtransport_priorityのパケット識別子を設定することによって、一つのエレメンタリーストリーム内において任意のパケットに優先度をつけることができる。

【0135】transport_scrambling_controlは、トランスポートストリームパケットペイロードのスクランブリングモードを示す2ビットのデータである。スクランブリングモードとは、ペイロードに格納されたデータがスクランブされているか否か及びそのスクランブルの種類を示すためのモードである。トランスポートストリームパケットヘッダ、およびアダプテーションフィールドは、スクランブルキーKsによってスクランブルされてはならないように規格化されている。よって、このtransport_scrambling_controlによって、トランスポートストリームパケットペイロードに格納されたデータがスクランブルされているか否かを判断することができる。

【0136】adaptation_field_controlは、このトランスポートストリームのパケットヘッダにアダプテーションフィールド及び／又はペイロードがくることを示す2ビットのデータである。具体的には、パケットヘッダにペイロードデータのみが配置される場合には、このadaptation_field_controlは「01」となり、パケットヘッダにアダプテーションフィールドのみが配置される場合には、このadaptation_field_controlは「10」となり、パケットヘッダにアダプテーションフィールドとペイロードとが配置される場合には、このadaptation_field_controlは「11」となる。

【0137】continuity_counterは、連続して伝送された同じPIDをもつパケットが、伝送途中で一部欠落又は捨てられか否かを示すためのデータである。具体的には、continuity_counterは、同一のPIDを有する各トランスポートストリームパケットごとに増加する4ビットのフィールドである。但し、このcontinuity_counterがカウントされるときは、パケットヘッダにアダプテーションフィールドが配置されている場合である。

【0138】adaptation_field() は、個別ストリームに関する付加情報やスタッフィングバイト等をオプションとして挿入するためのフィールドである。このアダプテーションフィールドによって、個別ストリームの動的な状態変化に関するあらゆる情報をデータと一緒に伝送することができる。

【0139】図19は、adaptation_field() のシンタックスを説明するための図である。このadaptation_field() は、adaptation_field_length、discontinuity_counter、random_access_indicator、elementary_stream_priority_indicator、OPCR_flag、splicing_point_flag、transport_private_data_flag、adaptation_field_extension_flag、program_clock_reference (PCR)、original_program_clock_reference (OPCR)、splice_countdown、transport_private_data_length、private_data、adaptation_field_extension_length、ltw_flag (legal_time_window_flag)、piecewise_rate_flag、及びseamless_splice_flag等の各種フィールドから構成されている。

【0140】adaptation_field_lengthは、このadaptation_field_lengthの次に続くアダプテーションフィールドのバイト数を示すデータである。adaptation_field_controlが「11」の場合には、adaptation_field_lengthは0から182ビットであって、adaptation_field_controlが「10」の場合には、adaptation_field_lengthは183ビットとなる。尚、トランスポートストリームのペイロードを満たすだけのエレメンタリーストリームが無い場合には、ビットを満たすためのスタッフィング処理が必要となる。

【0141】discontinuity_counterは、同じPIDを有する複数のパケットの途中において、システムクロックリファレンス(SCR)がリセットされ、システムクロックリファレンスが不連続になっているか否かを示すデータである。もし、システムクロックリファレンスが不連続の場合には、このdiscontinuity_counterは「1」となり、システムクロックリファレンスが連続している場合には、このdiscontinuity_counterは「0」となる。尚、このシステムクロックリファレンスとは、ビデオ及びオーディオのデコードするためのMPEGデコードにおいて、デコード側のシステムタイムクロックの値をエンコード側において意図したタイミングに設定するための参照情報である。

【0142】random_access_indicatorは、ビデオのシーケンスヘッダ又はオーディオのフレームの始まりを示すデータである。つまり、このrandom_access_indicatorは、データエレメントのランダムアクセスを行うときに、ビデオ又はオーディオのアクセスポイント(フレームの始まりのこと)であることを示すためのデータである。

【0143】elementary_stream_priority_indicatorは、同一のPIDを有するパケットにおいて、このトランスポートストリームパケットのペイロード中で伝送されるエレメンタリーストリームデータの優先度を示すデータである。例えば、エレメンタリーストリームがビデオデータが、そのビデオデータがイントラ符号化されている場合に、elementary_stream_priority_indicatorが「1」にセットされる。それに対して、インター符号化されているビデオデータを含んだトランスポートストリームのelementary_stream_priority_indicatorは、「0」にセットされる。

【0144】PCR_flagは、アダプテーションフィールド内にPCR(program_clock_reference)データが存在するか否かを示すデータである。アダプテーションフィールド内にPCRデータが存在する場合には、PCR_flagが「1」にセットされ、PCRデータが存在しない場合には、PCR_flagが「0」にセットされる。尚、このPCRデータとは、受信機側のデコードにおいて、伝送されたデータをデコードするデコード処理のタイミングを得るために使用されるデータである。

【0145】OPCR_flagは、アダプテーションフィールド内にOPCR(original_program_clock_reference)データが存在するか否かを示すデータである。アダプテーションフィールド内にOPCRデータが存在する場合には、OPCR_flagが「1」にセットされ、OPCRデータが存在しない場合には、OPCR_flagが「0」にセットされる。このOPCRデータとは、スプラインシング処理等によって、複数のオリジナルトランスポートストリームから1つのトランスポートストリームを再構築したときに使用されるデータであって、あるオリジナルトランスポートストリームのPCRデータを表わすデータである。

【0146】splicing_point_flagは、トランスポートレベルでの編集ポイント(スプライスポイント)を示すためのsplice_countdownがアダプテーションフィールド内に存在するか否かを示すデータである。アダプテーションフィールド内にsplice_countdownが存在する場合には、このsplicing_point_flagは「1」であって、アダプテーションフィールド内にsplice_countdownが存在する場合には、このsplicing_point_flagは「0」である。

【0147】transport_private_data_flagは、アダプテーションフィールド内に、任意のユーザデータを記述するためのプライベートが存在するか否かを示すためのデータである。アダプテーションフィールド内にプライベートが存在する場合には、このtransport_private_data_flagは「1」にセットされ、アダプテーションフィールド内にプライベートが存在しない場合には、このtransport_private_data_flagは「0」にセットされる。

【0148】adaptation_field_extension_flagは、アダプテーションフィールド内に、拡張フィールド存在するか否かを示すためのデータである。アダプテーションフィールド内に拡張フィールドが存在する場合には、このadaptation_field_extension_flagは「1」にセットされ、アダプテーションフィールド内に拡張フィールドが存在しない場合には、このadaptation_field_extension_flagは「0」にセットされる。

【0149】program_clock_reference(PCR)は、送信側とクロックの位相に対して受信機側のクロックの位相を同期させるときに参照する基準クロックである。このPCRデータには、トランスポートパケットが生成された時間が格納されている。このPCRは、33ビットのprogram_clock_reference_baseと9ビットのprogram_clock_reference_extensionとの42ビットから構成されるデータである。program_clock_reference_extensionによって0~299までのシステムクロックをカウントし、299から0にリセットされる際のキャリーによって、program_clock_reference_baseに1ビットを加算することによって、24時間をカウントすることができる。

【0150】original_program_clock_reference(OPC

R)は、あるトランスポートストリームから単一プログラムのトランスポートストリームを再構成するとき使用されるデータである。単一プログラムトランスポートストリームが完全に再構成された場合、このoriginal_program_clock_referenceはprogram_clock_referenceにコピーされる。

【0151】splice_countdownは、同一PIDのトランスポートストリームパケットにおいて、トランスポートストリームパケットレベルで編集可能(スプライシング処理可能)なポイントまでのパケットの数を示すデータである。従って、編集可能なスプライシングポイントのトランスポートストリームパケットでは、splice_countdownは「0」である。splice_countdownが「0」になるトランスポートパケットで、トランスポートストリームパケットペイロードの最終バイトは、符号化されたピクチャの最後のバイトとすることによって、スプライシング処理が可能となる。

【0152】このスプライシング処理とは、トランスポートレベルで行われる2つの異なるエレメンタリストリームを連結し、1つの新しいトランスポートストリームを生成する処理のことである。そして、スプライシング処理として、復号の不連続性を発生しないシームレススプライスと、復号の不連続性を引き起こすノンシームレススプライスとに分けることができる。符号の不連続性を発生しないとは、新しく後ろにつなげられたストリームのアクセスユニットの復号時間と、スプライス前の古いストリームのアクセスユニットの復号時間との間に矛盾が無いことを示し、符号の不連続性を発生するとは、新しく後ろにつなげられたストリームのアクセスユニットの復号時間に対して、スプライス前の古いストリームのアクセスユニットの復号時間の矛盾が生じることを示している。

【0153】transport_private_data_lengthは、アダプテーションフィールドにおけるプライベートデータのバイト数を示すデータである。

【0154】private_dataは、規格では特に規定されておらず、アダプテーションフィールドにおいて任意のユーザデータを記述することができるフィールドである。

【0155】adaptation_field_extension_lengthは、アダプテーションフィールドにおけるアダプテーションフィールドエクステンションのデータ長を示すデータである。

【0156】ltw_flag(legal_time_window_flag)は、アダプテーションフィールドにおいて表示ウィンドウのオフセット値を示すltw_offsetが存在するか否かを示すデータである。

【0157】piecewise_rate_flagは、アダプテーションフィールドにおいてpiecewise_rateが存在するか否かを示すデータである。

【0158】seamless_splice_flagは、スプライシング

ポイントが、通常のスプライシングポイントか、シームレススプライシングポイントであることを示すデータである。このseamless_splice_flagが「0」の時は、スプライシングポイントが通常のスプライシングポイントであることを示し、このseamless_splice_flagが「1」の時は、スプライシングポイントがシームレススプライシングポイントであることを示している。通常のスプライシングポイントとは、スプライシングポイントがPESパケットの区切りに存在する場合であって、このスプライシングポイントの直前のスプライシングパケットがアクセスユニットで終了し、次の同じPIDを有するトランスポートパケットがPESパケットのヘッダで開始している場合である。これに対して、シームレススプライシングポイントとは、PESパケットの途中でスプライシングポイントがある場合であって、新しく後ろにつなげられたストリームのアクセスユニットの復号時間と、スプライス前の古いストリームのアクセスユニットの復号時間との間に矛盾が無いようにするために、古いストリームの特性の一部を、新しいストリームの特性として使う場合である。

【0159】次に、本局30から伝送されたきたストリームSTOLDと地方局40において生成されたストリームSTNEWとをスプライシングするスプライシング処理について、図20から図23を参照して説明する。

【0160】図20は、図7において説明した地方局40の制御をよりわかりやすく説明するために、複数チャンネルうちのある1チャンネルのみを残して、他のチャンネルを省略した図である。本発明においては、スプライシング処理に関する実施例として3つのスプライシング処理に関する実施例を有している。以下に、第1、第2及び第3のスプライシング処理に関する実施例を順に説明する。

【0161】第1のスプライシング処理に関する実施例は、伝送プログラムの符号化ストリームSTOLDが本局30から伝送されてくる前に、新しく挿入されるコマーシャルCM'の符号化ストリームSTNEWが既に生成されている場合に行われるスプライシング処理に関する実施例である。つまり、伝送プログラムにおけるコマーシャルCMの符号化ストリームSTOLD部分に、既に前もって符号化されているコマーシャルCM1'のストリームを挿入する場合である。通常、コマーシャルというものは何度も繰り返して放映されるので、その度にコマーシャルのビデオデータを符号化するのでは効率的ではない。そこで、地方向けのコマーシャルCM1'のビデオデータを符号化し、その符号化ストリームSTNEWをストリームサーバ49に予め記憶しておく。そして、本局30から置換えられるコマーシャルCM1の符号化ストリームSTOLDが伝送されてきたときに、このストリームサーバ49から地方向けのコマーシャルCM1'の符号化ストリームSTNEWを再生することによって、同じ

コマーシャルを何度も符号化する処理を省くことができる。このような場合に、以下に具体的に説明する第1のスプライシング処理が行われる。

【0162】まず地方局40において、伝送プログラムのコマーシャルCM1の部分に置換えられる地方向けのコマーシャルCM1'を符号化し、符号化ストリームSTNEWをストリームサーバ49に格納する初期処理について説明する。放送システムコントローラ41は、伝送プログラムのコマーシャルCMの部分に置換えられるコマーシャルCM1'のビデオデータを再生するようにCMサーバ47を制御する。そして、エンコーダ481は、CMサーバ47から再生されたベースバンドのビデオデータを受取り、このビデオデータの各ピクチャの符号化難易度(Difficulty)Diをエンコーダコントローラ480に供給する。エンコーダコントローラ480は、図8において説明したエンコーダコントローラ350と同じように、エンコーダ481が適切な符号化ビットを発生するようにエンコーダ481に対してターゲットビットレートRiを供給する。エンコーダ481は、エンコーダコントローラ480から供給されたターゲットビットレートRiに基いて符号化処理を行なうことによって、最適なビットレートの符号化エレメンタリーストリームSTNEWを生成することができる。エンコーダ481から出力された符号化エレメンタリーストリームSTNEWは、ストリームサーバ49に供給される。ストリームサーバ49は、符号化エレメンタリーストリームをストリームの状態のまま、ランダムアクセス可能な記録媒体に記録する。これで、符号化ストリームSTNEWをストリームサーバ49に格納する初期処理は終了する。

【0163】次に、本局から伝送されてきた伝送プログラムの符号化ストリームSTOLDと、上述した初期処理によってストリームサーバ49に格納された符号化ストリームSTNEWとをスプライシングするスプライシング処理について説明する。

【0164】本局30から伝送されてきた符号化ストリームSTOLDは、ストリーム変換回路44においてトランスポートストリームの形式からエレメンタリーストリームの形式に変換される。エレメンタリーストリームの形式に変換された符号化ストリームSTOLDは、ストリームスプライサ50に供給される。

【0165】ストリームスプライサ50は、図20に示されているように、スプライスコントローラ500、スイッチ回路501、ストリーム解析回路502、ストリームプロセッサ503、及びスプライシング回路504を備えている。

【0166】この第1のスプライシング処理に関する実施例では、スプライスコントローラ500は、スイッチ回路501の入力端子を「a」に切換え、ストリームサーバ49から供給されたエレメンタリーストリームST

NEWをストリーム解析回路502に供給する。

【0167】ストリーム解析回路502は、符号化ストリームSTOLD及び符号化ストリームSTNEWのシンタックスを解析する回路である。具体的には、ストリーム解析回路502は、図10及び図15に開示された符号化ストリームのシンタックスから理解できるように、符号化ストリームSTOLD中に記述された32ビットのpicture_start_codeを探すことによって、ストリーム中においてピクチャヘッダに関する情報が記述された場所を把握する。次に、ストリーム解析回路502は、picture_start_codeの11ビット後から始まる3ビットのpicture_coding_typeを見つけることによって、ピクチャタイプを把握すると共に、この3ビットのpicture_coding_typeの次に記述された16ビットのvbv_delayから、符号化ピクチャのvbv_delayを把握することができる。

【0168】さらに、ストリーム解析回路502は、図10及び図15に開示された符号化ストリームのシンタックスから理解できるように、符号化ストリームSTOLD及び符号化ストリームSTNEW中に記述された32ビットのextension_start_codeを探すことによって、ストリーム中においてピクチャコーディングエクステンションに関する情報が記述された場所を把握する。次に、ストリーム解析回路502は、picture_start_codeの25ビット後から記述された1ビットのtop_field_firstと、そのtop_field_firstの6ビット後から記述されたrepeat_first_fieldとを探すことによって、符号化ピクチャのフレーム構造を把握することができる。例えば、符号化ピクチャの「top_field_first」が「1」である場合には、トップフィールドがボトムフィールドよりも時間的に早いフレーム構造であることを示し、「top_field_first」が「0」の場合には、トップフィールドがボトムフィールドよりも時間的に早いフレーム構造であることを示す。また、符号化ストリーム中のフラグ「top_field_first」が「0」であって且つ「repeat_first_field」が「1」の場合には、復号化時にトップフィールドからリピートフィールドが生成されるようなピクチャ構造であることを示し、符号化ストリーム中のフラグ「top_field_first」が「0」であって且つ「repeat_first_field」が「1」の場合には、復号化時にボトムフィールドからリピートフィールドが生成されるようなピクチャ構造を有していることを示している。

【0169】上述した、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldは、各ピクチャ毎に符号化ストリームから抽出され、スプライスコントローラ500に供給される。ストリーム解析回路502に供給されたエレメンタリーストリームSTOLD及びエレメンタリーストリームSTNEWは、そのままエレメンタリーストリームSTOLD及びエレメンタリーストリームSTNEWとしてストリームプロセッサ503に供給される。

【0170】さらに、ストリーム解析回路502は、供給されたストリームSTOLDとストリームSTNEWのビット数をカウントするためのカウンタを備えており、このカウント値と各ピクチャの発生ビット量とに基いて、各ピクチャ毎にVBVバッファのデータ残量をシュミレーションするようにしている。ストリーム解析回路502において演算された各ピクチャ毎のVBVバッファのデータ残量も、スプライスコントローラ500に供給される。

【0171】ストリームプロセッサ503は、ストリームSTOLDとストリームSTNEWとをスプライシングすることによって生成されるスプライドストリームSTSPLがシームレスなストリームとなるように、このストリームSTOLD及びストリームSTNEWのストリーム構造、データエレメント及びフラグを変更するための回路である。このストリームプロセッサ503の具体的な処理を、図21を参照して説明する。

【0172】図21(A)は、本局30から供給されたオリジナルストリームSTOLDと、そのストリームSTOLDのVBVバッファのデータ占有量の軌跡を示す図であって、図21(B)は、ストリームサーバ49に記憶された差し替えストリームSTNEWと、そのストリームSTNEWのVBVバッファのデータ占有量の軌跡を示す図であって、図21(B)は、スプライシングポイントSP1及びSP2において、ストリームSTOLDとストリームSTNEWとをスプライシングしたスプライドストリームSTSPLと、そのスプライドストリームSTSPLのVBVバッファのデータ占有量の軌跡を示す図である。尚、図21(A)において、DTS(デコーディングタイムスタンプ)を示し、SP1vbmは、VBVバッファのデータ占有量の軌跡上における第1のスプライシングポイントを示し、SP2vbmは、VBVバッファデータ占有量の軌跡上における第2のスプライシングポイントを示し、VO(I6)は、ピクチャB5がVBVバッファから引き出されたときにVBVバッファ上にバッファリングされているピクチャI6のデータ量を示し、GB(I6)は、ピクチャI6の発生ビット量を示し、VD(I6)は、ピクチャI6のvbm_delayの値を示し、VO(B7)は、ピクチャI6がVBVバッファから引き出されたときにVBVバッファ上にバッファリングされているピクチャB7のデータ量を示し、GB(B11)は、ピクチャB11の発生ビット量を示し、VD(I12)は、ピクチャI12のvbm_delayの値を示し、VO(I12)は、ピクチャB11がVBVバッファから引き出されたときにVBVバッファ上にバッファリングされているピクチャI12のデータ量を示している。また、図21(B)において、GB(I6')は、ピクチャI6'の発生ビット量を示し、VD(I6')は、ピクチャI6'のvbm_delayの値を示し、VO(I6')は、VBVにおける第1のスプライシングポイントSP1vbmにおけるピクチャI6'のVBVバッ

ファにバッファリングされているデータ量を示し、GB(B11')は、ピクチャB11'の発生ビット量を示し、VO(I12')は、VBVにおける第2のスプライシングポイントSP2vbmにおけるピクチャB12'のVBVバッファにバッファリングされているデータ量を示している。また、図21(C)において、GB(I6'')は、スプライドストリームSTSPLがシームレスなストリームとなるようにストリーム処理されたピクチャI6'の発生ビット量を示し、VD(I6'')は、ピクチャI6'のvbm_delayの値を示し、GB(B11'')は、スプライドストリームSTSPLがシームレスなストリームとなるようにストリーム処理されたピクチャB11'の発生ビット量を示している。

【0173】オリジナルストリームSTOLDは本局30において符号化されたストリームであって、差し替えストリームSTNEWは地方局40において符号化されたストリームであるので、ストリームSTOLDとストリームSTNEWとは、それぞれのビデオエンコーダで全く関係無く符号化されたストリームである。つまり、ストリームSTOLDにおける最初のピクチャI6のvbm_delayの値VD(I6)と、ストリームSTNEWにおける最初のピクチャI6'のvbm_delayの値VD(I6')とは同じ値ではない。つまり、このような場合には、VBVバッファにおけるストリームスプライスポイントSP1vbmのタイミングにおいて、オリジナルストリームSTOLDのVBVバッファのデータ占有量VO(I6)と、差し替えストリームSTNEWのVBVバッファのデータ占有量VO(I6')とは異なってしまう。

【0174】つまり、本発明の背景技術において説明したように、スプライスポイントSP1において、ストリームSTOLDとストリームSTNEWとを単純にスプライスしてしまうと、単純にスプライスされたストリームのVBVバッファのデータ占有量が不連続になるか又はオーバーフロー／アンダーフローしてしまう。

【0175】そこで、ストリームスプライサ50では、ストリーム解析回路502においてストリームSTOLD及びストリームSTNEWから抽出されたデータエレメントに基いて、スプライドストリームSTSPLがスプライスポイントにおいてシームレスなストリームとなるように、ストリームプロセッサ503において、供給されたストリームSTOLD及びストリームSTNEWのストリーム構造に関してストリーム処理を行なっている。その処理を以下に説明する。

【0176】スプライスコントローラ500は、ストリームSTOLDに関するデータエレメントとして、各ピクチャ毎に、picture_coding_type、vbm_delay、top_field_first及びrepeat_first_field等の情報、各ピクチャにおける発生ビット量、各ピクチャにおけるVBVバッファのデータ占有量をストリーム解析回路502から受け取る。図21においては、例えば、ピクチャI6にお

けるvbv_delayの値をVD(I6)と表わし、ピクチャI6における発生ビット量をGB(I6)と表わし、ピクチャI6におけるVBVバッファのデータ占有量をVO(I6)と表わしている。

【0177】次に、スプライスポイントSP1におけるスプライスコントローラ500とストリームプロセッサ503のVBVバッファに関する処理について説明する。

【0178】まず、スプライスコントローラ500は、スプライシングポイントSP1における、オリジナルストリームSTOLDのピクチャI6のvbv_delayの値VD(I6)と、差し替えストリームSTNEWのピクチャI6'のvbv_delayの値VD(I6')とが異なると判断した場合に、差し替えストリームSTNEW中に記述されたピクチャI6'のvbv_delayの値をVD(I6')からVD(I6)に書き換えるようにストリームプロセッサ503に指示を与える。

【0179】ストリームプロセッサ503は、スプライスコントローラ500からの指示に従って、差し替えストリームSTNEWのピクチャヘッダ中に記述された16ビットのvbv_delayの値をVD(I6')からVD(I6)に書き換える。

【0180】ここで、単に、差し替えストリームSTNEWにおけるvbv_delayの値をVD(I6')からVD(I6)に書

$$\begin{aligned} SB1[\text{byte}] &= \{ GB(I6'') - GB(I6') \} / 8 \\ &= \{ GB(I6) - GB(I6') + VO(B7) - VO(B7') \} / 8 \end{aligned} \quad (2)$$

【0182】スプライスコントローラ500は、上式(2)に従って演算したスタッフィングバイトSB1を、ストリームSTNEWの中に挿入するようにストリームプロセッサ503を制御する。

【0183】ストリームプロセッサ503は、スプライスコントローラ500からの指令に従って、スタッフィングバイトSB1を、ストリームSTNEWの中に記述する。スタッフィングバイトをストリーム中に記述する位置としては、符号化ストリームSTNEWのピクチャI6のピクチャヘッダのスタートコードの前が最も望ましいが、他のスタートコードの前であっても問題はない。

【0184】以上が、スプライスポイントSP1におけるスプライスコントローラ500とストリームプロセッサ503のVBVバッファに関する制御である。

【0185】次に、スプライスポイントSP2におけるスプライスコントローラ500とストリームプロセッサ503のVBVバッファに関する制御について説明する。

【0186】単に、スプライスポイントSP2において、ストリームSTNEWとストリームSTOLDとをスプライシングしたとすると、ストリームSTNEWの最後のピクチャB11'の発生ビット量GB(B11')が足りないの

き換え、この書き換えたvbv_delayに従ってVBVバッファからビットストリームを引き出そうとすると、ピクチャI6'の発生ビット量が足りないの、VBVバッファがアンダーフローしてしまう。そこで、スプライスコントローラ500は、差し替えストリームSTNEWのピクチャI6'の発生ビット量GB(I6')が、シームレスなスプライスドストリームSTSPのピクチャI6''の発生ビット量GB(I6'')になるように、差し替えストリームSTNEWのピクチャI6'のに対してスタッフィングバイトを挿入する処理を行なう。このスタッフィングバイトとは、「0」のダミービットから構成されるデータである。

【0181】スタッフィングバイトを挿入する処理を行うために、スプライスコントローラ500は、ストリームSTOLDにおけるピクチャI6及びピクチャB7に関する情報として受け取った発生ビット量GB(I6)、VBVバッファのデータ占有量VO(I6)、及びストリームSTNEWにおけるピクチャI6'に関する情報として受け取った発生ビット量GB(I6')、VBVバッファのデータ占有量VO(I6')を使用して、挿入すべきスタッフィングバイトのデータ量を演算する。具体的には、下記の式(2)に基いてスタッフィングバイトSB1[byte]が演算される。

で、ストリームSTNEWの最初のピクチャI12のVBVバッファのデータ占有量の軌跡と連続にならない。その結果、VBVバッファがアンダーフロー又はオーバーフローしてしまう。

【0187】そこで、スプライスコントローラ500は、VBVバッファにおけるスプライスポイントSP2 vbvにおいて、VBVバッファの軌跡が連続になるように、ストリームSTNEWの最後のピクチャB11'の発生符号量GB(B11')が、図21(C)のピクチャB11''の発生符号量GB(11'')となるように、ストリームSTNEW中にスタッフィングバイトを挿入する処理を行なう。

【0188】スタッフィングバイトを挿入する処理を行うために、スプライスコントローラ500は、ストリームSTOLDにおけるピクチャI12に関する情報として受け取ったVO(I12)、ストリームSTNEWの最後のピクチャ11'の発生ビット量GB(B11')、及びストリームSTNEWのピクチャ12'のVBVバッファのデータ占有量VO(I12')を使用して、挿入すべきスタッフィングバイトのデータ量を演算する。尚、具体的には、下記の式(2)に基いてスタッフィングバイトSB2[byte]が演算される。

$$SB2[\text{byte}] = \{ GB(B11'') - GB(B11') \} / 8$$

$$= \{ VO(I12') - VO(I12) \} / 8$$

(3)

【0189】尚、データ占有量 $VO(I12')$ とは、最後のピクチャ $B11'$ をVBVバッファから引き出した後のストリームSTNEWに関するVBVバッファのデータ占有量であると言い換えることができ、ストリームSTNEWのビット数をカウントすることによってVBVの軌跡を把握しているストリーム解析回路502によって、容易にこのデータ占有量 $VO(I12')$ 検出することができる。

【0190】このスプライスコントローラ500は、上式(3)に従って演算したスタッフィングバイトSB2を、ストリームSTNEW中に挿入するようにストリームプロセッサ503を制御する。

【0191】ストリームプロセッサ503は、スプライスコントローラ500からの指令に従って、スタッフィングバイトSB2を、ストリームSTNEWのピクチャ $B11'$ に関する情報として記述する。スタッフィングバイトをストリーム中に記述する位置としては、符号化ストリームSTNEWのピクチャ $B11'$ のピクチャヘッダのスタートコードの前が最も望ましい。

【0192】以上が、スプライスポイントSP2におけるスプライスコントローラ500とストリームプロセッサ503のVBVバッファに関する制御である。

【0193】図22を参照して、スプライスポイントSP1におけるスプライスコントローラ500とストリームプロセッサ503の top_field_first 及び $repeat_first_field$ 等のフラグに関する第1の処理例を説明する。

【0194】図22(A)は、本局30において制作されたプログラム1とコマーシャルCM1とプログラム2から構成されるテレビ番組PGOLDのフレーム構造と、そのテレビ番組PGOLDを符号化したときの符号化ストリームSTOLDを示している図である。図22(B)は、地方局40において制作された差し替えコマーシャルCM1'のフレーム構造と、その差し替えコマーシャルCM1'を符号化したときの符号化ストリームSTNEWを示している図である。図22(C)は、オリジナルストリームSTOLDと差し替えストリームSTNEWとをスプライシングしたときに生成されるスプライスドストリームSTSPLと、そのスプライスドストリームSTSPLをデコードしたときのフレーム構造を示している図である。

【0195】スプライスコントローラ500は、ストリーム解析回路502から供給されたストリームSTOLDにおけるコマーシャルCM1の各ピクチャの top_field_first と、差し替えストリームSTNEWにおけるコマーシャルCM1'の top_field_first とを比較する。もし、ストリームSTOLDにおける top_field_first と、差し替えストリームSTNEWにおける top_field_first とが一致するのであれば、フィールド構造が同じであるので、to

p_field_first 及び $repeat_first_field$ 等のフラグに関する処理は必要ない。しかし、図22に示すように、オリジナルコマーシャルCM1の top_field_first が「0」であって、差し替えコマーシャルCM1'の top_field_first が「1」である場合には、図6において説明したようなフィールドの不連続及び重複という問題が発生する。

【0196】そこで、本発明のストリームスプライサ50は、スプライシング処理によってフィールドが欠けたり重複したりするようなMPEGストリーム違反のストリームが生成されないように、スプライシングポイント付近のピクチャの top_field_first 及び $repeat_first_field$ を書き換えるようにしている。

【0197】図22に示された例においては、スプライスコントローラ500は、トップフィールドT4とボトムフィールドB4とからフレームが構成されるピクチャP3の $repeat_first_field$ を0から1に書き換えるようにストリームプロセッサ503を制御する。さらに、スプライスコントローラ500は、スプライスポイントSP2においてシームレスなストリームとなるように、トップフィールドt10とボトムフィールドb11からフレームが構成されるピクチャP9'の $repeat_first_field$ を0から1に書き換えるようにストリームプロセッサ503を制御する。また、スプライスコントローラ500は、ピクチャP9'の $repeat_first_field$ を書き換えたことによって、コマーシャルCM1'がオリジナルコマーシャルCM1に対して1フレーム時間だけシフトしたので、プログラム2において最初にディスプレイに表示されるピクチャB13を、ストリームSTOLD中から削除するようにストリームプロセッサ503を制御する。

【0198】ストリームプロセッサ503は、スプライスコントローラ500の指示に基いて、オリジナルストリームSTOLDにおいてピクチャP3に関する $picture_coding_extension$ のスタートコードを探し、その中の $repeat_first_field$ の値を0から1に書き換える。よって、このように $repeat_first_field$ の値が書き換えられたピクチャP3をデコードすると、リピートフィールドB4'が生成されるので、スプライスポイントSP1においてフィールドが連続することになる。同様に、ストリームプロセッサ503は、差し替えストリームSTNEWにおいてピクチャP9'に関する $picture_coding_extension$ のスタートコードを探し、その中の $repeat_first_field$ の値を0から1に書き換える。このように $repeat_first_field$ の値が書き換えられたピクチャP9'をデコードすると、リピートフィールドt10'が生成されるので、スプライスポイントSP2においてフィールドが連続することになる。また、ストリームプロセッサ5

03、オリジナルストリームSTOLDにおいてピクチャB13に関するデータエレメントが記述された部分を、ストリームSTOLD中から削除又はヌルデータに置き換える。

【0199】図23は、図22において説明したtop_field_first及びrepeat_first_field等のフラグに関する処理の他の処理例を表わしたものである。図23を参照して、スプライスポイントSP1及びSP2におけるスプライスコントローラ500とストリームプロセッサ503のtop_field_first及びrepeat_first_field等のフラグに関する第2の処理例を説明する。

【0200】図23に示された処理例においては、スプライスコントローラ500は、スプライスポイントSP1におけるプログラム1とコマーシャルCM1'とのつなぎめにおけるフィールドが連続するように、トップフィールドb5とボトムフィールドb6とから構成されるピクチャB7'のtop_field_firstを1から0に書き換えると共に、ピクチャB7'のrepeat_first_fieldを0から1に書き換えるようにストリームプロセッサ503を制御する。さらに、スプライスコントローラ500は、スプライスポイントSP2においてコマーシャルCM1'とプログラム2のつなぎめにおけるフィールドが連続するように、トップフィールドT11とボトムフィールドB11から構成されるピクチャB13のtop_field_firstを1から0に書き換えるようにストリームプロセッサ503を制御する。さらに、スプライスコントローラ500は、トップフィールドT12とボトムフィールドB12から構成されるピクチャB14のtop_field_firstを1から0に書き換えると共に、repeat_first_fieldを1から0に書き換えるようにストリームプロセッサ503を制御する。

【0201】ストリームプロセッサ503は、スプライスコントローラ500の制御に従って、差し替えストリームSTNEWにおいてピクチャB7'に関するpicture_coding_extensionのスタートコードを探し、そのストリーム中のtop_field_firstを1から0に書き換えると共に、repeat_first_fieldを0から1に書き換える。よって、このようにtop_field_first及びrepeat_first_fieldの値が書き換えられたピクチャB7'をデコードすると、ボトムフィールドb6の表示時間が1フレーム分だけシフトし、かつ、リピートフィールドb6'が生成されるので、スプライスポイントSP1においてフィールドが連続することになる。同様に、ストリームプロセッサ503は、オリジナルストリームSTOLDにおいてピクチャB13に関するpicture_coding_extensionのスタートコードを探し、その中のtop_field_firstを1から0に書き換える。さらに、ストリームプロセッサ503は、オリジナルストリームSTOLDにおいてピクチャB14に関するtop_field_firstを1から0に書き換えると共に、repeat_first_fieldを1から0に書き換える。

よって、このようにtop_field_first及びrepeat_first_fieldの値が書き換えられたピクチャB13及びB14をデコードすると、ボトムフィールドB11及びB12の表示時間が1フレーム分だけシフトするので、スプライスポイントSP2においてフィールドが連続することになる。

【0202】ここで、図22に示された第1の処理例と、図23に示された第2の処理例とを比較すると、図22(C)から理解できるように、差し替えられたコマーシャルCM1'の最初に表示されるピクチャB7'が、オリジナルコマーシャルCM1の最初に表示されるピクチャB7よりも、1フィールドだけずれているので、差し替えられたコマーシャルCM1'の表示タイミングが1フィールドだけ遅れてしまう。表示が1フィールド遅れた程度では、人間の目にはほとんどその遅れは分からない。しかしながら、放送局ではクライアント会社からのコマーシャルを放映することによって収入を得ているので、プログラム1等の本編を放映することよりもコマーシャルを遅れなく正確に放映することが要求される場合がある。このような正確な表示時間が要求される場合には、図23に示された第2の処理例が有効である。図23に示された第2の処理例のように、ピクチャB7'のtop_field_first及びrepeat_first_fieldの値を書き換えることによって、差し替えられたコマーシャルCM1'の最初のピクチャB7'を、オリジナルコマーシャルCM1の最初に表示されるピクチャに対して遅れ無く正確に表示することができる。

【0203】つまり、ストリームプロセッサ503から出力されたストリームSTNEWのVBVバッファのデータ占有量の軌跡は、ストリームSTOLDのVBVバッファのデータ占有量の軌跡と整合性が取れており、且つ、フィールドパターン/フレームパターンに関する整合性が取れている。よって、スプライスコントローラ500からの制御信号に基いてスプライシング回路504のスイッチング動作を制御することによって、スプライスポイントSP1で、ストリームSTOLDの後にストリームSTNEWを接続し、スプライスポイントSP2で、ストリームSTNEWの後にストリームSTOLDを接続することによって生成されたストリームであって、そのスプライスポイントSP1及びSP2において、VBVバッファのデータ占有量の軌跡が連続しており、且つフィールドパターン/フレームパターンが連続になっているスプライスドストリームSTSPLが生成される。

【0204】次に、第2のスプライシング処理に関する実施例について説明する。第2のスプライシング処理に関する実施例は、伝送プログラムの符号化ストリームSTOLDが本局30から伝送されてきたときに、新しく挿入されるコマーシャルCM'を符号化して符号化ストリームSTNEWを生成する場合に行われるスプライシング処理に関する実施例である。つまり、本局30から伝送

されてきた伝送プログラムの符号化ストリームSTOLDを解析し、その解析結果に基いて、新しく挿入されるコマーシャルCM1'を符号化するという方法である。

【0205】まず、本局30から伝送されてきた符号化ストリームSTOLDは、ストリーム変換回路44においてトランスポートストリームの形式からエレメンタリーストリームの形式に変換される。エレメンタリーストリームの形式に変換された符号化ストリームSTOLDは、ストリームスプライサ50のストリーム解析回路502に供給される。

【0206】ストリームスプライサ50のストリーム解析回路502は、符号化ストリームSTOLDのストリームシンタックスを解析するための回路である。この第2のスプライシング処理に関する実施例においては、このストリーム解析回路502は、符号化ストリームSTOLDのシンタックスの解析のみを行い、差し替えストリームSTNEWのシンタックスの解析は行なわない。

【0207】具体的には、まず、ストリーム解析回路502は、図10及び図15に開示された符号化ストリームのシンタックスから理解できるように、オリジナルストリームSTOLD中に記述された32ビットのpicture_start_codeを探すことによって、ストリーム中においてピクチャヘッダに関する情報が記述された場所を把握する。次に、ストリーム解析回路502は、picture_start_codeの11ビット後から始まる3ビットのpicture_coding_typeを見つけることによって、ピクチャタイプを把握すると共に、この3ビットのpicture_coding_typeの次に記述された16ビットのvbv_delayから、符号化ピクチャのvbv_delayを把握することができる。

【0208】さらに、ストリーム解析回路502は、図10及び図15に開示された符号化ストリームのシンタックスから理解できるように、符号化ストリームSTOLD中に記述された32ビットのextension_start_codeを探すことによって、ストリーム中においてピクチャコーディングエクステンションに関する情報が記述された場所を把握する。次に、ストリーム解析回路502は、picture_start_codeの25ビット後から記述された1ビットのtop_field_firstと、そのtop_field_firstの6ビット後から記述されたrepeat_first_fieldとを探すことによって、符号化ピクチャのフレーム構造を把握することができる。

【0209】ストリーム解析回路502は、オリジナルストリームSTOLD中から各ピクチャ毎に抽出された、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_field等のデータエレメントを、スプライサコントローラ500を介して、放送システムコントローラ41に供給する。尚、オリジナルストリームSTOLDの全ピクチャのデータエレメントを送る必要は無く、伝送プログラムにおけるコマーシャルCM1に対応するピクチャのpicture_coding_type、vbv_delay、top_fiel

d_first及びrepeat_first_field等のデータエレメントのみであっても良い。

【0210】次に、放送システムコントローラ41は、伝送プログラムのコマーシャルCMの部分に差し替えられるコマーシャルCM1'のビデオデータを再生するようにCMサーバ47を制御する。さらに、放送システムコントローラ41は、オリジナルストリームSTOLDから抽出されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを、エンコードブロック48のエンコードコントローラ480に供給する。

【0211】エンコードコントローラ480は、放送システムコントローラ41から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを使用して、差し替えコマーシャルCM1'のベースバンドビデオデータをエンコードするようにエンコード481を制御する。すなわち、オリジナルコマーシャルCM1の符号化ストリームSTOLDのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldと、差し替えコマーシャルCM1'を符号化したストリームSTNEWのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldとが全く同じになるように、差し替えコマーシャルCM1'を符号化する。その結果、オリジナルコマーシャルCM1の符号化ストリームSTOLDのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldと、全く同じpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを有した符号化したストリームSTNEWが生成される。

【0212】スプライサコントローラ501は、スイッチ回路501の入力端子を「b」に切換え、エンコード481から出力されたエレメンタリーストリームSTNEWをストリーム解析回路502に供給する。この第2のスプライシング処理に関する実施例においては、このストリーム解析回路502は、符号化ストリームSTOLDのシンタックスの解析のみを行い、差し替えストリームSTNEWのシンタックスの解析は行なわないので、ストリームSTNEWは、ストリーム解析回路502において解析処理されずにそのまま出力される。

【0213】ストリームプロセッサ503は、この第2のスプライシング処理に関する実施例では、ストリーム解析回路502から出力されたストリームSTOLD及びストリームSTNEWにおけるデータエレメントを変更するようなストリーム処理は必要ないので、ストリームSTOLDとストリームSTNEWとのフレーム同期を合わせる同期合わせ処理（フレームシンクロナイゼーション）のみを行なう。具体的には、このストリームプロセッサ503は、数フレーム分のFIFOバッファを有しており、差し替えストリームSTNEWがエンコード481から出力されるまで、このFIFOバッファにストリームSTOLDをバッファリングしておくことによって、スト

リームSTOLDとストリームSTNEWのフレーム同期を合わせることができる。フレーム同期合わせ処理されたストリームSTOLDとストリームSTNEWは、スプライシング回路504に供給される。

【0214】スプライスコントローラ500は、スプライシングポイントSP1においてストリームSTOLDの後にストリームSTNEWが接続され、スプライシングポイントSP2においてストリームSTNEWの次にストリームSTOLDが接続されるように、スプライシング回路504のスイッチングを制御する。その結果、スプライシング回路504からは、スプライスドストリームSTSPLが出力される。

【0215】単純にスプライシング回路504において、ストリームSTOLDとストリームSTNEWとをスイッチングしているだけであるが、スプライスドストリームSTSPLのVBVバッファのデータ占有量の軌跡は連続となっており、且つ、スプライスポイントにおけるフレームパターンも連続している。なぜなら、オリジナルストリームSTOLDをストリームのシンタックスの解析結果を基に、ストリームSTNEWを符号化処理しているので、オリジナルストリームSTOLDに対して整合性の取れたストリームSTNEWを生成しているため、スプライスドストリームSTSPLのVBVバッファの軌跡は、オリジナルストリームSTOLDのVBVバッファの軌跡と全く同じであって、生成されたスプライスドストリームSTSPLのフレーム構造は、オリジナルストリームSTOLDのフレーム構造と全く同じであるからである。

【0216】従って、この第2の実施例によれば、本局から伝送されてきたオリジナル符号化ストリームSTOLDのシンタックスを解析し、その解析結果に応じて、符号化ストリームSTOLDと同じストリーム構造及び符号化パラメータを有するように、差し替えコマmercialCM1'を符号化しているので、それぞれ別に生成された、符号化ストリームSTOLDと符号化ストリームSTNEWをスプライシングする際に、符号化ストリームSTOLDと符号化ストリームSTNEWとの整合性を合わせることとなる簡単にスプライシングを行なうことができ、その結果、MPEG規格に準じた且つシームレスなスプライスドストリームSTSPLを生成することができる。

【0217】次に、第3のスプライシング処理に関する実施例について説明する。第3のスプライシング処理に関する実施例は、オリジナルコマmercialCM1の符号化ストリームSTOLD及び、差し替えられるコマmercialCM1'の符号化ストリームSTNEWを生成する前に、予め、オリジナルコマmercialCM1の符号化ストリームSTOLDと差し替えコマmercialCM1'の符号化ストリームSTNEWを符号化するための符号化パラメータを決定しておき、その決っている符号化パラメータに基づいてオリジナルコマmercialCM1及び差し替えコマmercialCM1'を符号化するという処理である。

例えば、この符号化パラメータとは、既に説明した、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどや発生ビット量等によって示される情報である。

【0218】まず、本局30において、オリジナルコマmercialCM1を符号化するための符号化パラメータとして、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを決定する。本局30の放送システムコントローラ31は、その符号化パラメータを、MPEGエンコーダブロック35のエンコーダコントローラ350に供給すると共に、通信回線を使用して各地方局40の放送システムコントローラ41にも供給する。

【0219】エンコーダコントローラ350は、放送システムコントローラ31から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどの符号化パラメータを使用してオリジナルコマmercialCM1のビデオデータを符号化するようにビデオエンコーダ351-1Vを制御する。すなわち、ビデオエンコーダ351-1Vから出力された符号化ストリームSTOLDは、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどの符号化パラメータに基づいているストリームである。

【0220】ビデオエンコーダ351-1Vから出力された符号化ストリームSTOLDは、マルチプレクサ36及び変調回路37を介して地方局40に供給される。

【0221】一方、地方局40は、本局30の放送システムコントローラ31から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_field等の符号化パラメータを、エンコーダブロック48のエンコーダコントローラ480に供給する。

【0222】エンコーダコントローラ480は、放送システムコントローラ41から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを使用して、差し替えコマmercialCM1'のベースバンドビデオデータをエンコードするようにエンコーダ481を制御する。すなわち、オリジナルコマmercialCM1の符号化ストリームSTOLDのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldと、全く同じpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを有した符号化したストリームSTNEWが生成される。

【0223】スプライスコントローラ501は、スイッチ回路501の入力端子を「b」に切換え、エンコーダ481から出力されたエレメンタリストリームSTNEWをストリーム解析回路502に供給する。この第3のスプライシング処理に関する実施例においては、このストリーム解析回路502は、符号化ストリームSTOLD及び符号化ストリームSTNEWのシンタックスの解析は行わない。

【0224】ストリームプロセッサ503は、この第2のスプライシング処理に関する実施例では、ストリーム解析回路502から出力されたストリームSTOLD及びストリームSTNEWにおけるデータエレメントを変更するようなストリーム処理は必要ないので、ストリームSTOLDとストリームSTNEWとのフレーム同期を合わせる同期合わせ処理（フレームシンクロナイゼーション）のみを行なう。具体的には、このストリームプロセッサ503は、数フレーム分のFIFOバッファを有しており、差し替えストリームSTNEWがエンコード481から出力されるまで、このFIFOバッファにストリームSTOLDをバッファリングしておくことによって、ストリームSTOLDとストリームSTNEWのフレーム同期を合わせることができる。フレーム同期合わせ処理されたストリームSTOLDとストリームSTNEWは、スプライシング回路504に供給される。

【0225】スプライスコントローラ500は、スプライシングポイントSP1においてストリームSTOLDの後にストリームSTNEWが接続され、スプライシングポイントSP2においてストリームSTNEWの次にストリームSTOLDが接続されるように、スプライシング回路504のスイッチングを制御する。その結果、スプライシング回路504からは、スプライスドストリームSTSPLが出力される。

【0226】単純にスプライシング回路504において、ストリームSTOLDとストリームSTNEWとをスイッチングしているだけであるが、スプライスドストリームSTSPLのVBVバッファのデータ占有量の軌跡は連続となっており、且つ、スプライスポイントにおけるフレームパターンも連続している。なぜなら、予め本局30の放送システムコントローラ31において決定された、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどの符号化パラメータを使用して、オリジナルコマーシャルCM1及び差し替えコマーシャルCM1'が符号化されているからである。

【0227】従って、この第3の実施例によれば、本局及び地方局間において、予め符号化パラメータを決定しておき、その決められた符号化パラメータに基いて本局においてオリジナルコマーシャルCM1を符号化した符号化ストリームSTOLDを生成し、さらに、その決められた符号化パラメータに基いて本局において差し替えコマーシャルCM1'を符号化した符号化ストリームSTNEWを生成するようにしているので、それぞれ別に生成された、符号化ストリームSTOLDと符号化ストリームSTNEWをスプライシングする際に、符号化ストリームSTOLDと符号化ストリームSTNEWとの整合性を合わせることとなる簡単にスプライシングを行なうことができ、その結果、MPEG規格に準じた且つシームレスなスプライスドストリームSTSPLを生成することができる。

【0228】

【発明の効果】請求項1の符号化ストリームスプライシング装置及び請求項21の符号化ストリーム生成装置によれば、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第1の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリームとがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシング手段とを備えることによって、スプライシングされたストリームのVBVバッファのデータ占有量の軌跡が連続であって、且つ、VBVバッファは破綻しないシームレスなスプライシング処理を実現することができる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【0229】また、請求項11の符号化ストリームスプライシング方法及び請求項22の符号化ストリーム生成方法によれば、第1の符号化ストリームのシンタックスを解析することによって、第1の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第1の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第1の符号化ストリームと第2の符号化ストリームとがシームレスに接続されるように、第2の符号化ストリームの符号化パラメータを変更し、第1の符号化ストリームと符号化パラメータが変更された第2の符号化ストリームとをスプライシングするスプライシングステップとを行なうことによって、スプライシングされたストリームのVBVバッファのデータ占有量の軌跡が連続であって、且つ、VBVバッファは破綻しないシームレスなスプライシング処理を実現することができる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【0230】請求項23の情報処理装置及び請求項27の情報処理方法によれば、受信したデータの一部を、他のデータと入れ替え、その入れ替える他のデータを符号化し、受信したデータを、他のデータと整合が取れるように変更するようにしたので、不整合などを生じることなくデータの入れ替えが可能となる。

【図面の簡単な説明】

【図1】一般的なテレビジョン放送システムを説明するための図である。

【図2】地方局におけるテレビ番組の編集処理を説明するための図である。

【図3】MPEG規格を用いた一般的なデジタル伝送システムを説明するための図である。

【図4】VBVバッファの制御について説明するための図である。

【図5】2:3プルダウン処理を説明するための図である。

【図6】符号化ストリームをスプライシング処理したときに発生する問題点を説明するための図である。

【図7】本発明の実施の形態に係る符号化ストリームスプライシング装置を含むデジタル放送システムを全体構成を示す図である。

【図8】本局30のMPEGエンコーダブロック35及び地方局40のエンコーダブロックの構成を詳細に説明するためのブロック図である。

【図9】ストリーム変換回路においてエレメンタリーストリームからトランスポートストリームを生成するときの処理を説明するための図である。

【図10】MPEGのビデオエレメンタリーストリームのシーケンスのシンタックスを説明するための図である。

【図11】シーケンスヘッダ(sequence_header)のシンタックスを説明するための図である。

【図12】シーケンスエクステンション(sequence_extension)のシンタックスを説明するための図である。

【図13】エクステンションアンドユーザデータ(extension_and_user_data)のシンタックスを説明するための図である。

【図14】グループオブピクチャヘッダ(group_of_picture_header)のシンタックスを説明するための図である。

【図15】ピクチャヘッダ(picture_header)のシンタ

ックスを説明するための図である。

【図16】ピクチャコーディングエクステンション(picture_coding_extension)のシンタックスを説明するための図である。

【図17】ピクチャヘッダ(picture_data)のシンタックスを説明するための図である。

【図18】トランスポートストリームパケットのシンタックスを説明するための図である。

【図19】アダプテーションフィールド(adaptation_field)のシンタックスを説明するための図である。

【図20】地方局の簡単な構成及びストリームスプライサの構成を説明するための図である。

【図21】ストリームスプライサのVBVバッファに関する処理を説明するための図である。

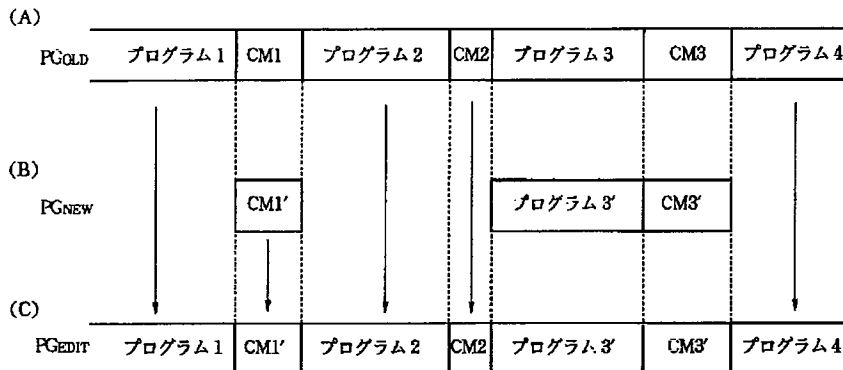
【図22】ストリームスプライサのtop_field_first及びrepeat_first_fieldに関する第1の処理例を説明するための図である。

【図23】ストリームスプライサのtop_field_first及びrepeat_first_fieldに関する第2の処理例を説明するための図である。

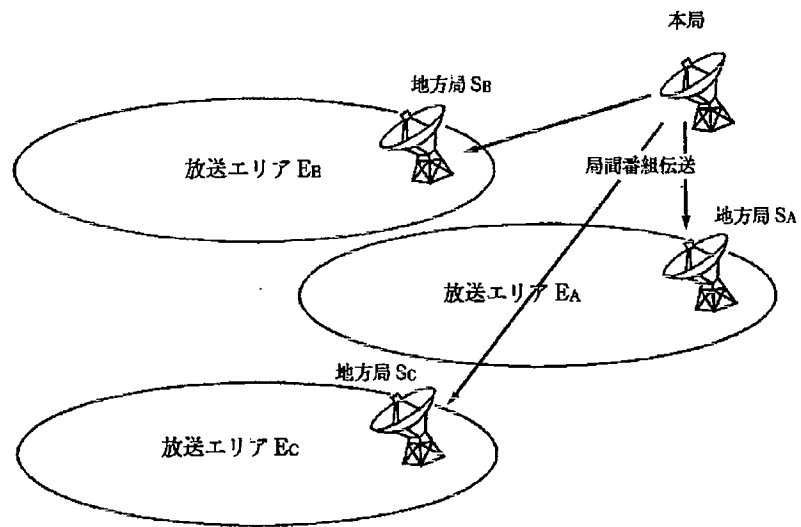
【符号の説明】

31 放送システムコントローラ, 32 素材サーバ, 33 CMサーバ, 34 マトリックススイッチャブロック, 35 MPEGエンコーダブロック, 36 マルチプレクサ, 37 変調回路, 41 放送システムコントローラ, 42 復調回路, 43 デマルチプレクサ, 44 ストリーム変換回路, 46 素材サーバ, 47 CMサーバ, 48 エンコーダブロック, 49 ストリームサーバ, 50 ストリームスプライサ, 51 ストリーム変換回路, 52 マルチプレクサ, 53 変調回路, 61, 62家庭

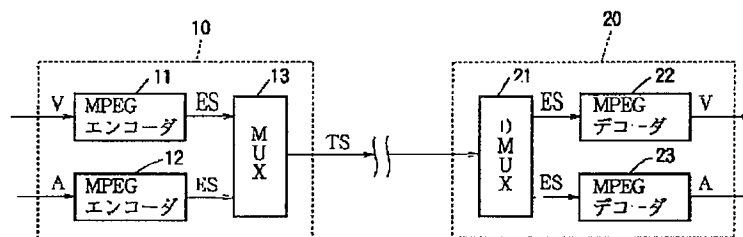
【図2】



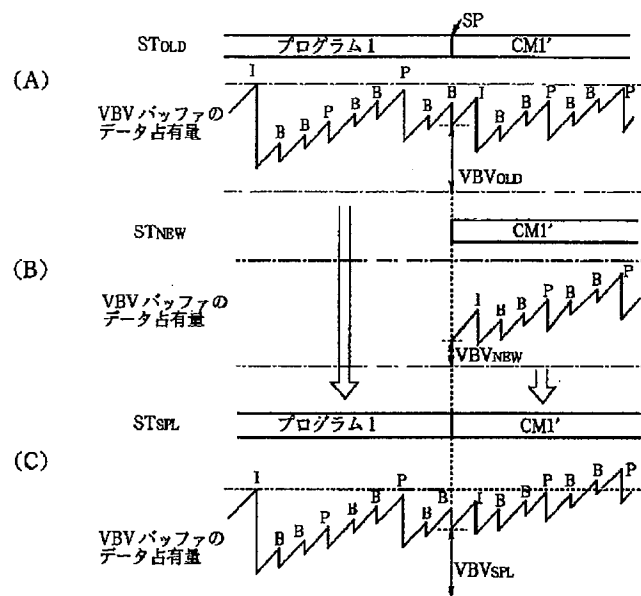
【図1】



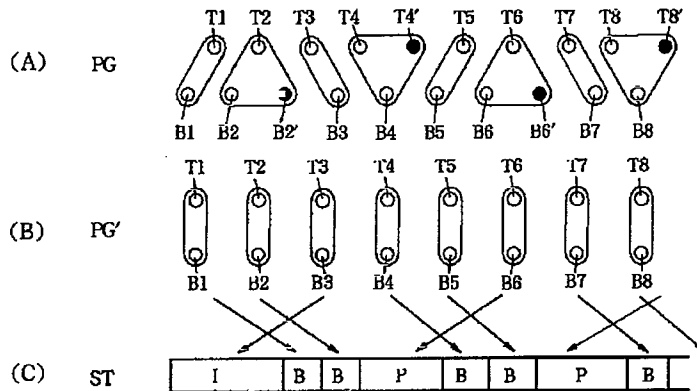
【図3】



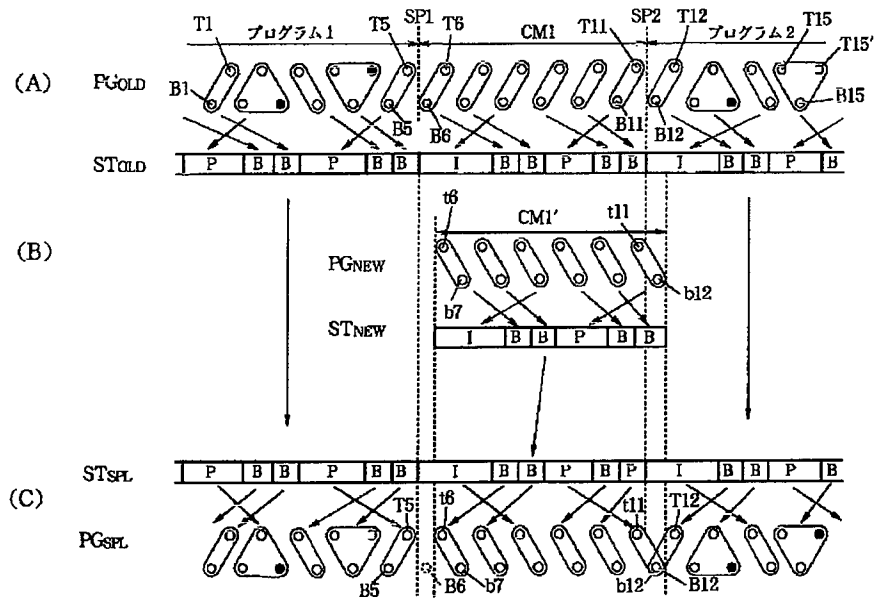
【図4】



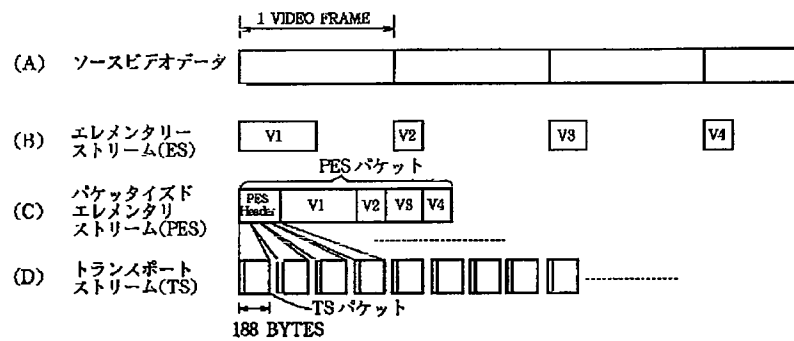
【図5】



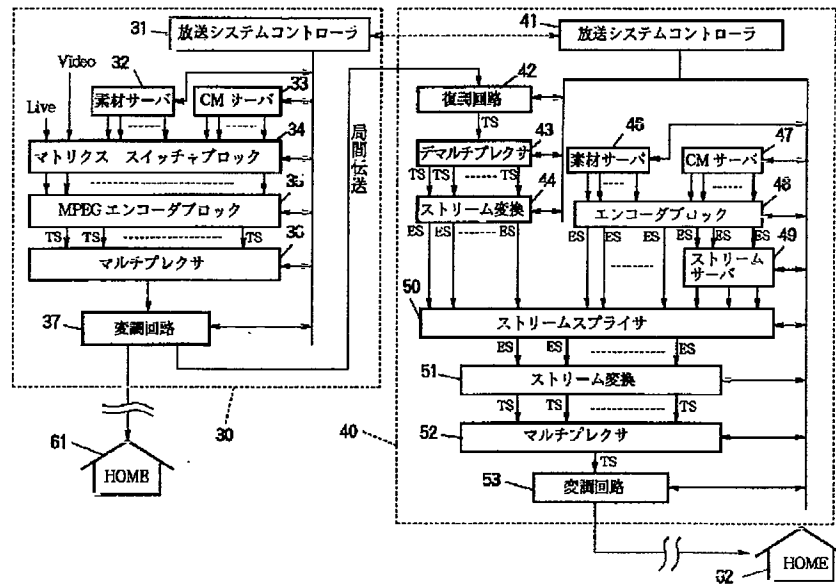
【図6】



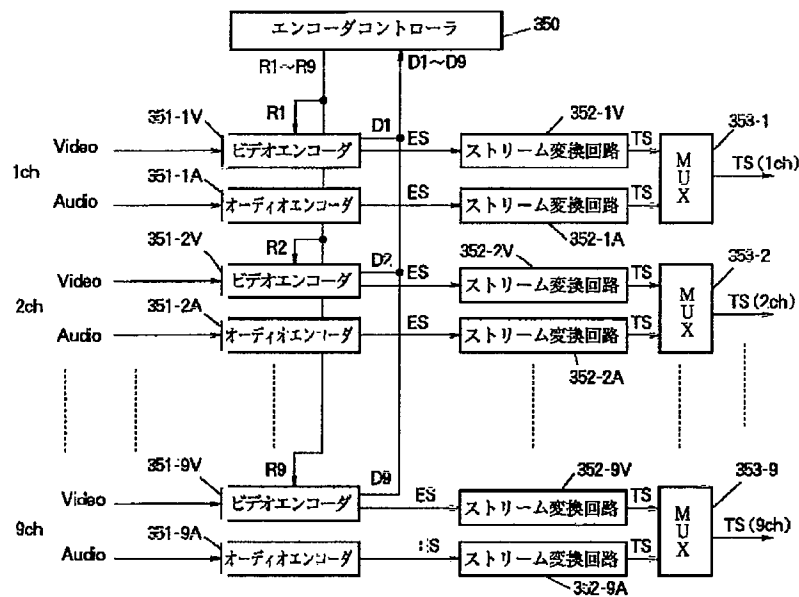
【図9】



【図7】



【図8】



【図17】

ピクチャデータ

picture_data()	ビット数	ニーモニック
do {		
slice()		
} while (nextbits() == slice_start_code)		
next_start_code()		
}		

【図10】

video_sequence() {	ビット数	ニーモニック
next_start_code()		
sequence_header()		
sequence_extension()		
do {		
extension_and_user_data(0)		
do {		
if (nextbits() == group_start_code) {		
group_of_pictures_header()		
extension_and_user_data()		
}		
picture_header()		
picture_coding_extension()		
extension_and_user_data(2)		
picture_data()		
} while ((nextbits() == picture_start_code)		
(nextbits() == group_start_code))		
if (nextbits() != sequence_end_code) {		
sequence_header()		
sequence_extension()		
}		
} while (nextbits() != sequence_end_code)		
sequence_end_code	32	bslbf
}		

【図13】

拡張データおよびユーザデータ

extension_and_user_data(i) {	ビット数	ニーモニック
while (((i!=1) its() && (nextbits() extension_start_code))		
(nextbits() == user_data_start_code)) {		
if (nextbits() == extension_start_code)		
extension_data(i)		
if (nextbits() == user_data_start_code)		
user_data()		
}		
}		

【図11】

シーケンスヘッダ

sequence_header() {	ビット数	ニーモニック
sequence_header_code	32	bslbf
horizontal_size_value	12	uimsbf
vertical_size_value	12	uimsbf
aspect_ratio_information	4	uimsbf
frame_rate_code	4	uimsbf
bit_rate_value	18	uimsbf
marker_bit	1	"1"
vbv_buffer_size_value	10	uimsbf
constrained_parameters_flag	1	
load_intra_quantise_matrix	1	
if (load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	
if (load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8*64	uimsbf
next_start_code()		
}		

【図12】

シーケンス拡張

sequence_extension() {	ビット	ニーモニック
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_identifier	8	uimsbf
progressive_sequence	1	uimsbf
chroma_format	2	uimsbf
horizontal_size_extension	2	uimsbf
vertical_size_extension	2	uimsbf
bit_rate_extension	12	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_extension	8	uimsbf
low_delay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code()		
}		

【図14】

グループオブピクチャヘッダ

group_of_picture_header() {	ビット数	ニーモニック
group_start_code	32	bslbf
time_code	25	bslbf
closed_gop	1	uimsbf
broken_link	1	uimsbf
next_start_code()		
}		

【図15】

ピクチャヘッダ

picture_header() {	ビット数	ニーモニック
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if (picture_coding_type==2 picture_coding_type==3) {		
full_pel_forward_vector	1	
forward_f_code	3	uimsbf
}		
if (picture_coding_type==3) {		
full_pel_backward_vector	1	
backward_f_code	3	uimsbf
}		
while (nextbits()=='1') {		
extra_bit_picture /* with the value "1" */	1	uimsbf
extra_information_picture	8	
}		
extra_bit_picture /* with the value "0" */	1	uimsbf
next_start_code() {		
}		

【図16】

ピクチャコーディング拡張

picture_coding_extension() {	ビット数	シーモニック
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
f_code[0][0] /* forward horizontal */	4	uimsbf
f_code[0][1] /* forward vertical */	4	uimsbf
f_code[1][0] /* backward horizontal */	4	uimsbf
f_code[1][1] /* backward vertical */	4	uimsbf
intra_dc_precision	2	uimsbf
picture_structure	2	uimsbf
top__field__first	1	uimsbf
frame_pred_frame_dct	1	uimsbf
concealment_motion_vectors	1	uimsbf
q_scale_type	1	uimsbf
intra_vic_format	1	uimsbf
alternate_scan	1	uimsbf
repeat_first_field	1	uimsbf
chroma_420_type	1	uimsbf
progressive_frame	1	uimsbf
composite_display_flag	1	uimsbf
if (composite_display_flag) {		
v_axis	1	uimsbf
field_sequence	3	uimsbf
sub_carrier	1	uimsbf
burst_amplitude	7	uimsbf
sub_carrier_phase	8	uimsbf
}		
next_start_code()		
}		

【図19】

シンタックス	ビット数	ニーモニック
adaptation_field() {		
adaptation_field_length	8	unimbf
if(adaptation_field_length>0) {		
discontinuity_indicator	1	bslbf
random_access_indicator	1	bslbf
elementary_stream_priority_indicator	1	bslbf
PCR_flag	1	bslbf
OPCR_flag	1	bslbf
splicing_point_flag	1	bslbf
transport_private_data_flag	1	bslbf
adaptation_field_extension_flag	1	bslbf
if(PCR_flag=='1') {		
program_clock_reference_base	33	uimbsbf
reserved	6	bslbf
program_clock_reference_extension	9	uimbsbf
}		
if(OPCR_flag=='1') {		
original_program_clock_reference_base	33	uimbsbf
reserved	6	bslbf
original_program_clock_reference_extension	9	uimbsbf
}		
if(splicing_point_flag=='1') {		
splice_countdown	8	tcimbsbf
}		
if(transport_private_data_flag=='1') {		
transport_private_data_length	8	uimbsbf
for(i=0; i<transport_private_data_length; i++) {		
private_data_byte	8	bslbf
}		
}		
if(adaptation_field_extension_flag=='1') {		
adaptation_field_extension_length	8	unimbsbf
ltw_flag	1	bslbf
piecewise_rate_flag	1	bslbf
seamless_splice_flag	1	bslbf
reserved	5	bslbf
if(ltw_flag=='1') {		
ltw_valid_flag	1	bslbf
ltw_offset	15	unimbsbf
}		
if(piecewise_rate_flag=='1') {		
reserved	2	bslbf
piecewise_rate	22	uimbsbf
}		
if(seamless_splice_flag=='1') {		
splice_type	4	bslbf
DTS_next_AU[32..30]	3	bslbf
marker_bit	1	bslbf
DTS_next_AU[29..15]	15	bslbf
marker_bit	1	bslbf
DTS_next_AU[14..0]	15	bslbf
marker_bit	1	bslbf
}		
for(i=0; i<N; i++) {		
reserved	8	bslbf
}		
for(i=0; i<N; i++) {		
stuffing_byte	8	bslbf
}		
}		

【図23】

